Carreira técnica no universo da programação

Desvendando depois do sênior e além

Casa do Código | alura

WILLIAM OLIVEIRA

Sumário

- ISBN
- Sobre o livro
- Agradecimentos
- Sobre o autor
- Parte 1: Contextualização
  - 1 Entendendo a trilha técnica
  - 2 Como acontece a evolução profissional em engenharia de software
  - 3 O papel de especialista
  - 4 Os diferentes perfis das pessoas especialistas
- Parte 2: Prática
  - 5 Entendendo a parte prática da especialização
  - 6 Colocando os pés no chão
  - 7 A consolidação como sênior
  - 8 Colocando em prática
  - 9 Conclusão
  - 10 Referências

ISBN

Impresso: 978-85-5519-348-4

Digital: 978-85-5519-349-1

Dados Internacionais de Catalogação na Publicação (CIP)
(Câmara Brasileira do Livro, SP, Brasil)

Oliveira, William
Carreira técnica no universo da programação : desvendando depois do sênior e além / William Oliveira. -- São Paulo : AOVS Sistemas de Informática, 2023.

ISBN 978-85-5519-348-4

1. Capacitação profissional 2. Ciência da computação - Estudo e ensino 3. Programação (Computadores) 4. Programação (Computadores) - Estudo e ensino 5. Software - Desenvolvimento I. Título.

23-170333 CDD-005.1

Índices para catálogo sistemático:

1. Programação de computadores : Processamento de dados 005.1

Eliane de Freitas Leite - Bibliotecária - CRB 8/8415

Caso você deseje submeter alguma errata ou sugestão, acesse http://erratas.casadocodigo.com.br.

Sobre o livro

A carreira em desenvolvimento de software passou por muitas mudanças nos últimos anos. Antes éramos chamados de analistas programadores; depois, somente programadores; aí surgiram os títulos de *webmaster*, *front-end* e *back-end*, *mobile developer*, *desktop developer* e por aí vai.

Mas a mudança não aconteceu somente no título dos nossos cargos: a nossa trilha profissional também mudou. Antigamente, a única trilha profissional possível era: estágio ou assistente → júnior → pleno → sênior → **gerência** → **diretoria**. Isso fazia com que toda pessoa desenvolvedora seguisse o caminho direto para a gestão de pessoas e, depois, para a administração de uma empresa.

Mas e quem não tem interesse nesse futuro? O que seria de quem tem o foco central em software/sistemas/tecnologia e não se sente bem gerenciando pessoas? Pensando não somente nas nossas ambições pessoais, mas também nas necessidades do mercado em que estamos inseridos, se todo mundo está caminhando para a gestão de pessoas, quem está cuidando das estruturas técnicas de sustentação dos produtos digitais que nasceram com o crescimento da indústria? Alguém deve estar sobrecarregado(a) ou fazendo um trabalho pela metade.

Foi para suprir essa necessidade de mercado — a de possuir produtos digitais mais sustentáveis para o longo prazo, mais rentáveis para as organizações e mais eficientes no que fazem — que nasceu a **trilha técnica no universo da programação**, uma opção de carreira para aquelas pessoas que querem trabalhar com um escopo mais amplo dentro de uma organização, porém cuidando dos sistemas e não das tarefas relacionadas a pessoas. É para isso que existe a carreira como especialista em engenharia de software.

Neste livro, você vai encontrar uma visão geral do mercado de trabalho como especialista. Você vai entender como algumas empresas funcionam e como esse cargo se conecta com a

organização, onde a pessoa especialista se encaixa na estratégia de crescimento de uma empresa e os diferentes perfis e papéis de pessoas especialistas. Mergulhando na parte prática, você vai aprender como trabalhar de maneira eficiente, como conseguir uma promoção ou pleitear uma vaga, além de algumas dicas que ajudam pessoas neste nível profissional a equilibrar suas tarefas e funções.

A quem se destina este livro

A ideia central deste livro é racionalizar e materializar a carreira de especialista. Algumas pessoas já conhecem esse caminho, principalmente pelo contato com empresas estrangeiras ou com *big techs*, mas muitas ainda têm dúvidas sobre ele.

Você vai aproveitar muito bem este livro se for uma pessoa desenvolvedora em nível pleno e sênior, que deseja descobrir quais são os próximos possíveis passos que você pode seguir; se estiver em um cargo de gestão e precisa direcionar alguém que deseja se tornar especialista; ou se estiver na diretoria ou em cargos mais altos de uma organização e deseja entender a trilha para especialista e montar esse caminho na sua empresa.

Este livro conecta experiências pessoais do autor com fontes e referências do mercado digital. Portanto, será um grande guia para quem deseja seguir nessa trilha ou se espelhar em casos reais para organizar sua empresa.

Como este livro está estruturado

O livro está dividido em duas partes: *Contextualização* e *Prática*. Partimos da ideia de que algumas pessoas não conhecem a trilha de especialista em engenharia de software e, por isso, vamos primeiro contextualizar o assunto para ter um melhor aproveitamento da leitura. Em seguida, mergulharemos na parte prática da atuação como especialista, desde a consolidação profissional até as dúvidas mais recorrentes de profissionais da área.

Ambas as partes deste livro apresentam a visão do mercado e são permeadas por experiências pessoais, não só do autor, mas também de profissionais que são referência na nossa área e que foram entrevistados(as) para trazer suas vivências sobre os diversos aspectos discutidos nos capítulos.

No capítulo 1, temos uma introdução geral sobre a trilha técnica. Eu conto um pouco como foi a minha experiência até aqui e o que vamos encontrar no decorrer do livro.

No capítulo 2, vamos tentar padronizar minimamente o que acontece na indústria para entender os diferentes níveis e cargos na área de engenharia de software, além de conhecermos o conceito de carreira em Y.

No capítulo 3, vamos entender qual é o real papel de uma pessoa especialista em engenharia de software, conhecer os níveis e títulos da carreira de especialista e visualizar onde essas pessoas se encaixam na estratégia organizacional.

O capítulo 4 é destinado a refletirmos sobre os diferentes perfis profissionais e como podemos explorar isso para conseguir equilíbrio entre nossas ambições individuais e os objetivos da organização.

O capítulo 5 é uma introdução à parte prática da atuação como especialista em engenharia de software.

Antes de mergulharmos no cargo de especialista, o capítulo 6 apresenta alguns conselhos para não cairmos nas armadilhas da ansiedade ou de uma promoção prematura.

O capítulo 7 traz sugestões para que a pessoa que deseja seguir como especialista se consolide e ganhe conhecimento antes de pleitear o próximo nível.

Colocando tudo isso em prática, o capítulo 8 é um pouco mais denso, mas fornece dicas realmente valiosas e práticas de como atuar como especialista dentro de uma organização, e o capítulo 9 é a conclusão da nossa jornada.

Agradecimentos

Eu agradeço à comunidade de programação, por sempre me apoiar em minhas iniciativas, e, em especial, ao Elton Soares, que foi meu diretor e que, no momento em que eu estava confuso sobre a minha atuação, com muita paciência e empatia, me ajudou a encontrar o meu caminho.

Sobre o autor

Desenvolvedor de software desde 2014, William Oliveira é palestrante, escritor, podcaster, patinador e tibiano nas horas vagas.

Nascido na periferia de São Bernardo do Campo, no grande ABC Paulista, encontrou o caminho da programação através das comunidades de tecnologia por volta de 2007. Iniciou no universo da programação como desenvolvedor full-stack, atuando com a linguagem PHP e migrando para o front-end quando o JavaScript ganhou grandes responsabilidades em aplicações de larga escala, o que deu à pessoa desenvolvedora de software a possibilidade de pensar em interfaces web como aplicações, não mais como páginas estáticas.

Apaixonado por *open source*, software livre, conhecimento aberto, comunidades e afins, fundou o FEMUG-ABC, um grupo de *meetups* de desenvolvedores front-end que depois deu origem ao maior evento de tecnologia do grande ABC, o ABCDev. Também iniciou o Training Center, uma comunidade inclusiva focada em ajudar pessoas a entrarem na área de desenvolvimento de software, e, posteriormente, criou o *perifaCode*, uma organização sem fins lucrativos cujo foco é a inclusão de pessoas periféricas no mercado de trabalho.

É voluntário no núcleo de tecnologia do MTST, onde ensina tecnologia para o movimento que mais ajuda pessoas em situação de vulnerabilidade social do país.

Acredita fortemente que só a educação pode mudar a sociedade em que vivemos e, por isso, compartilha seus pensamentos e conhecimento pela internet afora. Desde 2014 contribui na vida das pessoas e as incentiva a entrar na área de programação através do seu blog pessoal, `woliveiras.com.br` , além de dar palestras, fazer workshops e participar de comunidades.

Como uma pessoa vinda da favela brasileira, entendo bem a dificuldade das pessoas menos privilegiadas, os grupos sub-representados, e busco apoiar a diversidade e a inclusão de minorias na área de programação de computadores.

Parte 1: Contextualização

Capítulo 1
Entendendo a trilha técnica

Quando iniciei na área de programação, eu não fazia ideia alguma de como seria o futuro profissional de uma pessoa que trabalha nessa área. Como são os primeiros meses? Como são os primeiros anos? Como seriam os próximos 10 anos trabalhando como desenvolvedor de software?

O que eu conhecia era o clássico modelo organizacional no qual uma pessoa consegue emprego, trabalha nesta empresa por 20/30 anos e, durante sua carreira, vai subindo de nível até o cargo de gerência ou acima, como diretoria ou presidência, e, normalmente, quem diz para onde esse indivíduo vai é alguém que está acima dele na hierarquia organizacional. Não existiam mentorias ou livros sobre carreira em programação, e eu **acredito** que isso também não acontecia para nenhuma outra profissão — afinal de contas, *mentoria profissional* é algo bem recente no mercado de trabalho: surgiu por volta da década de 70 como uma ferramenta de desenvolvimento profissional nas empresas e só recebeu este nome por volta dos anos 90.

Como o tempo não espera nosso período de preparação, logo eu estava trabalhando com tecnologia e não imaginava o que fazer da minha vida. Tanto que eu iniciei como técnico em informática, depois trabalhei como analista de suporte e hoje sou engenheiro de software. Porém eu tive a sorte de entrar em uma empresa que nos ensinava a fazer um tal de **plano de carreira**. Ali eu aprendi que eu poderia ser analista 1, 2 ou 3 antes de pleitear uma vaga para me tornar **gestor**.

Lá eu escrevi em uma folha de papel qual seria o meu futuro profissional, como eu me enxergava dali a seis meses, um ano, três

anos e cinco anos de carreira. Naquela época, eu trabalhava com infraestrutura, na parte de administração de sistemas, e meu sonho pessoal era me tornar desenvolvedor de software, portanto meu plano de carreira foi esse: me tornar desenvolvedor em dois anos.

Como parte do exercício de reflexão, eu precisava procurar na internet quais seriam os possíveis níveis profissionais dentro de uma hierarquia organizacional para uma pessoa desenvolvedora de software. E assim eu o fiz. Descobri que existem certas nomenclaturas para os cargos dentro de engenharia que são diferentes do mercado tradicional a que eu estava acostumado.

Normalmente nós acessamos uma empresa no cargo de auxiliar, depois nos tornamos analistas, buscamos o cargo de supervisão e, em seguida, vamos para gestão; depois disso, só o tempo nos dirá se conseguiremos (ou queremos) alcançar um nível de diretoria ou presidência de uma empresa.

Para a engenharia de software, eu encontrei os seguintes cargos: estagiário(a) ou auxiliar, júnior, pleno, sênior, gerente de engenharia, diretor(a) de engenharia. Em alguns lugares eu encontrei cargos como supervisor(a), coordenador(a) e superintendente, mas eles não se repetiram tantas vezes para que eu prestasse atenção, então eu preferi seguir com os títulos mais comuns.

Com base nos níveis descobertos, eu deveria iniciar meu plano de carreira para cinco anos. Eu nunca me perguntei por que cinco anos, mas, depois desse tempo trabalhando na área, entendi que é porque acontecem muitas coisas dentro de um espaço tão longo de tempo, e muitas coisas podem mudar, portanto as pessoas que nos ensinaram a planejar para esta faixa queriam nos manter com uma certa margem para que pudéssemos de planos no meio do caminho. Afinal de contas, nós não trabalhamos somente cinco anos na vida: no Brasil, atualmente, nós levamos pelo menos 30 anos de contribuição para podermos nos aposentar — e, normalmente, o brasileiro não para de trabalhar depois da aposentadoria.

Como eu ainda não atuava na área de programação, preferi escrever um plano de seis meses para antes do início real do meu plano de carreira de me tornar desenvolvedor em dois anos. Esse seria o tempo de transição da área de infraestrutura para a de desenvolvimento de software e, neste período, eu procuraria um emprego como auxiliar ou *junior software engineer*. Depois disso, eu focaria em me tornar desenvolvedor pleno em três anos e, dois anos depois, eu deveria estar no cargo de *senior software engineer*. Pensei no tempo de cinco anos para me tornar sênior porque foi o que percebi na maioria dos currículos de pessoas sêniores que encontrei pela internet.

Este projeto deu certo. Eu consegui seguir meu plano de carreira e finalmente cheguei ao cargo de sênior. Mas o que vem depois? Qual seria meu projeto para os próximos cinco anos? Eu deveria virar gestor? Em algumas empresas isso nem é uma questão, a única progressão profissional possível é o caminho da gestão.

Eu já trabalhei em lugares onde o único caminho profissional é o da gestão, e o que eu senti foi que algumas pessoas só estavam ali porque são obrigadas, porque é a única maneira de aumentar o salário e o impacto delas dentro da organização. Isso torna a experiência dessas pessoas algo ruim, pois elas lutam consigo mesmas todos os dias para focar em pessoas, sendo que o que lhes atrai a atenção o tempo todo é a parte técnica. Isso faz com que, na minha experiência profissional, eu tenha me encontrado com gestores que eram excelentes pessoas tecnicamente, mas péssimos em nos auxiliar além do código. Não nos ajudavam a direcionar nossa carreira, não nos apoiavam em momentos em que era necessário subir discussões a nível organizacional (como quando envolve dinheiro da empresa, contratar mais pessoas etc.). Ou seja, eram excelentes **programadores** e péssimos **gestores**.

> Será que essas pessoas deveriam realmente ter virado gestoras ou deveria existir um modo de elas continuarem sendo excelentes programadoras pelo resto de sua vida laboral?

O resultado dessa pressão para que seguissem o caminho de gestão trouxe uma rotatividade alta nas equipes de engenharia (pessoas saindo e entrando na empresa mais rápido do que em outras do mesmo setor), além do clima desagradável de pessoas estressadas o dia inteiro por microgerenciamento ou por não terem suas necessidades profissionais básicas atendidas.

Algo que eu sempre tive muita certeza em minha carreira profissional é que eu não gosto de lidar com a parte de gestão de pessoas dos lugares e projetos em que eu trabalhei. Eu adoro liderar, porém não gosto nem um pouco da parte organizacional do trabalho com pessoas. Mas qual outro caminho eu poderia seguir senão o de me tornar gestor? Essa foi uma dúvida que me estressou por muito tempo. O que eu realmente gosto é de estar próximo dos projetos de código, das decisões de arquitetura e das escolhas técnicas relacionadas à engenharia de software.

Eu também gosto de coisas não relacionadas com a parte técnica, como fazer reuniões *one on one*, onde conversamos sobre carreira, ajudar as pessoas a terem um ambiente de trabalho mais saudável, auxiliar para que as pessoas consigam ter dias mais produtivos etc. Porém essas não são as atribuições pelas quais eu gostaria de ser cobrado na minha avaliação de performance semestral ou que minha liderança me pedisse para fazer isso o tempo todo. O que eu gosto de lidar é com disponibilidade de software, segurança, acessibilidade e qualidade de código. Essas são as áreas em que eu gosto de investir minha energia e no que eu acredito ser bom.

Cheguei a essa conclusão depois de conversar com várias pessoas que atuam em cargos de gestão. Algumas com pouco tempo de experiência, outras em nível de diretoria. Algumas dessas pessoas

inclusive me direcionaram sobre o que eu poderia fazer além de ser gestor. Foi aí que eu descobri que, hoje em dia, existem ramificações na engenharia de software para comportar pessoas que querem continuar atuando na parte técnica depois de sênior.

Algumas empresas chamam o cargo depois de sênior de *especialista*, outras chamam de *expert*, outras chamam de *staff software engineer* (para gourmetizar a coisa e deixar mais atrativo no LinkedIn). Em todos os casos, estamos falando de um caminho onde você continua sua evolução profissional e aumenta seu impacto organizacional ainda mais depois de sênior, porém cuidando de software, não de pessoas. Mas eu também encontrei pessoas pelo caminho que gostariam de continuar no cargo de sênior para o resto da vida, e descobri que existem empresas em que até isso também é possível. Ou seja, se você é como eu e não quer atuar na área de gestão, existem caminhos possíveis para o nosso perfil.

A partir das próximas páginas, vamos explorar o que é ser um(a) especialista em engenharia de software e de que maneira você, que já é uma pessoa sênior ou está quase lá, pode se tornar especialista e o que fazer para se manter nesse cargo. Vamos falar sobre o que é esperado de você, como se preparar para isso e como você pode se manter atualizado(a) e relevante para o mercado. O conteúdo não será relevante somente para nós, que desejamos seguir como especialistas, mas também para quem tem curiosidade e ainda não sabe para onde ir, além de pessoas gestoras que desejam ajudar seus times a se desenvolverem.

Espero que as próximas páginas possam ajudar você a tirar essas dúvidas e inspirar seus próximos passos!

Capítulo 2
Como acontece a evolução profissional em engenharia de software

> "Nem toda pessoa com perfil de liderança deseja um cargo no qual sua atribuição principal será liderar."

Eu ouvi essa frase de uma pessoa gestora enquanto conversava sobre minha indecisão entre me tornar gestor ou continuar onde eu estava. A minha sorte foi o fato de que o meu momento de virada chegou enquanto a indústria se adaptava a pessoas que não queriam sair da área técnica.

Você já deve ter tido contato com o termo *carreira em Y*. Se não teve, agora é a hora! Eu só tive contato com esse conceito depois de seis anos atuando como desenvolvedor, e foi algo que mudou minha vida. Foi nessa conversa com esse gestor que aprendi que, a partir daquele instante, eu não era mais obrigado a fazer uma troca tão grande de escopo profissional mudando da área técnica para cuidar de pessoas. Eu poderia seguir como técnico pelo resto da minha vida.

Foi nesse momento que eu decidi o que fazer da minha carreira. Por isso, eu gostaria de compartilhar neste capítulo o que me foi apresentado, não somente naquela conversa, mas também depois, em outras empresas, palestras e livros que li sobre progressão de carreira em Y.

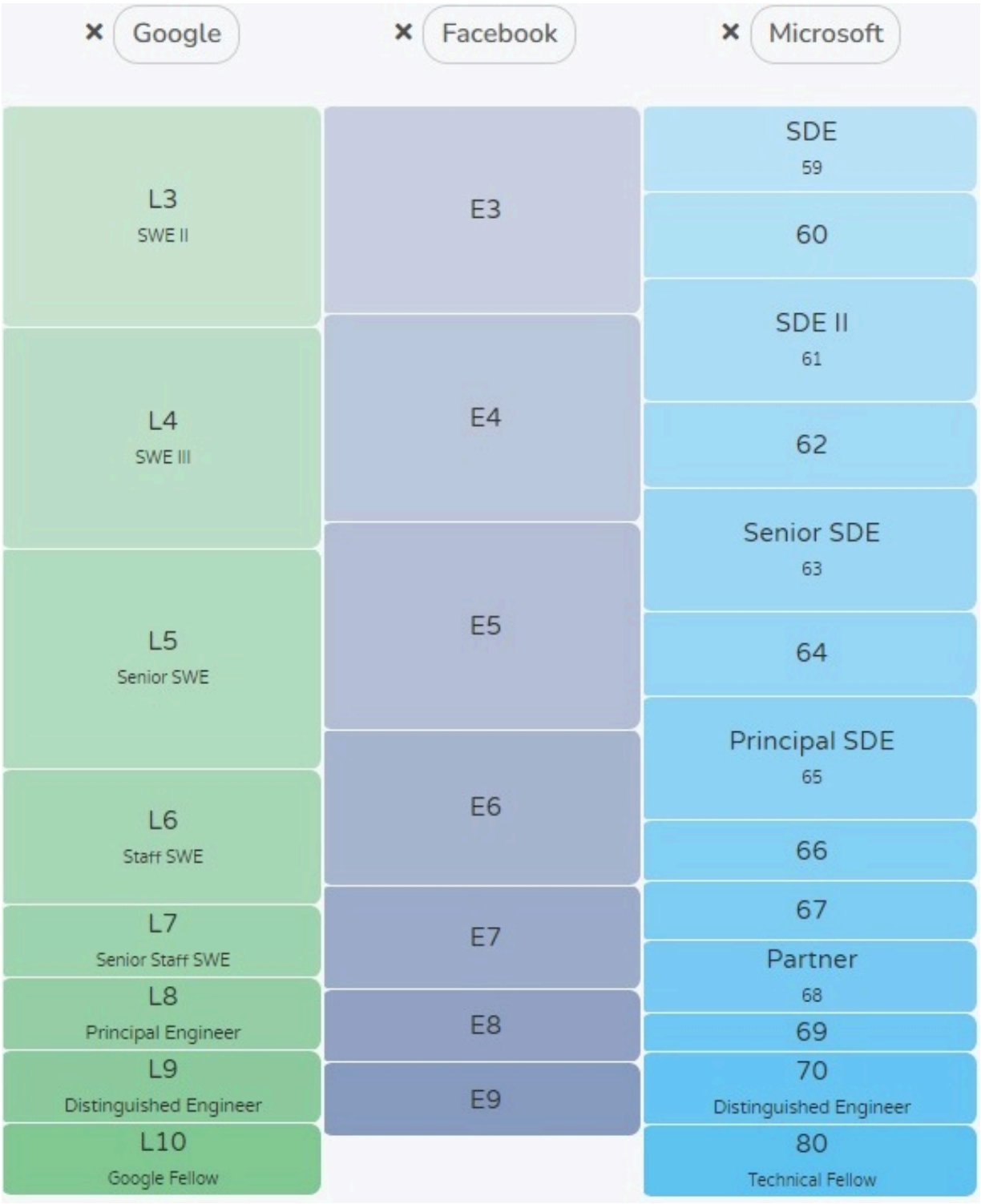
Vamos começar pelos diferentes níveis profissionais e depois falaremos sobre a tal da **carreira em Y** em si, para que, assim, você também possa tomar uma decisão assertiva para o seu futuro.

2.1 A dificuldade de entender os níveis e cargos em engenharia de software

Diferente de outras áreas, como medicina ou direito, nós não temos uma nomenclatura universal para os cargos em engenharia de software. Isso acontece porque a nossa área é muito nova e ainda está em constante mudança, além do fato de que outras engenharias são regulamentadas e precisam seguir um padrão. Em nosso caso, não temos uma regulamentação forte — o que, inclusive, possibilitou a minha entrada nessa área sem cursar uma faculdade.

Por conta disso, cada empresa costuma chamar seus profissionais como achar melhor e também criar novos níveis de acordo com a necessidade de promoção.

Para exemplificar melhor o que eu estou falando, extraí a imagem a seguir do site *levels.fyi*, no qual estão catalogados cargos e salários de empresas de tecnologia.



Percebemos que cada empresa utiliza uma nomenclatura e pode colocar os níveis como se encaixarem melhor em sua hierarquia organizacional. No exemplo dos cargos do Google, temos o nível L3,

que é o nível de "entrada na empresa", enquanto no Facebook o nível de entrada é o E3 e, na Microsoft, é o SDE **59**!

O que vou tentar fazer neste livro é compilar, para o mais próximo possível da simplificação, os cargos por títulos que encontramos nas vagas de emprego e/ou nos perfis de pessoas com esses cargos em redes sociais, por exemplo. Seria impossível catalogar todos os cargos e níveis que existem no mercado, mas acredito que, com o que vou apresentar aqui, você consiga se localizar e entender o que está acontecendo na sua carreira. Com isso você pode se preparar para encarar qualquer título desses ou, se for uma pessoa gestora, planejar como implementar algo similar em sua empresa.

2.2 Os diferentes níveis profissionais

Em nossa trilha profissional, normalmente vamos atuar em alguns cargos com títulos similares. A nomenclatura pode variar entre empresas, e vamos conversar sobre essas diferenças mais à frente na leitura. Por ora, vamos focar na evolução profissional mais padrão possível, para abranger o máximo de pessoas; depois, veremos alguns casos muito específicos e que destoam do que o mercado pratica.

Para entender o caminho que estamos seguindo, é importante que tiremos uma reflexão a respeito da responsabilidade de cada cargo. Qual é o papel de uma pessoa júnior, de alguém em nível pleno e de outra pessoa em nível sênior dentro de um time?

O nosso nível profissional, na maioria das vezes, está atrelado à independência e à autonomia dentro de um time e ao impacto na empresa ou organização/projeto em que atuamos. Digo organização/projeto porque nem toda pessoa desenvolvedora de software está trabalhando em uma empresa com organização de negócios tradicional. Nós temos pessoas que só trabalham com open

source e outras que são consultoras ou freelancers, cujo modelo de contratação não é o padrão com que estamos habituados.

Entretanto, quanto mais independente você é em sua função, maior está o seu nível profissional. Isso é um consenso entre lideranças em várias indústrias, não só em engenharia de software. Quando iniciamos, temos uma pessoa que nos guia nas tarefas, normalmente alguém em um nível maior que o nosso. Depois de algum tempo atuando na área, começamos a não depender tanto de ajuda e passamos a performar mais, precisando agora focar em qualidade e aumentar a nossa maestria na área de atuação. Com isso, teremos autonomia suficiente para propor melhorias, ensinar outras pessoas e mudar a direção em que uma equipe caminha.

Para clarear nossa mente, vamos imaginar exemplos específicos por nível profissional dentro de engenharia de software, começando pelos níveis júnior, pleno e sênior. O título de especialista ou sênior+ será aprofundado nos próximos capítulos.

É importante entendermos o que vem em cada situação antes da virada para especialista, para nos situarmos exatamente onde estamos agora. Em cada cargo, eu deixo uma explicação do que encontramos no mercado e também uma tabela com exemplos de atribuições — lembrando que todos os exemplos aqui citados podem funcionar diferente na sua empresa ou não ser exatamente o que exemplifiquei.

Júnior

Vamos assumir que a pessoa júnior de quem eu estou falando é alguém que acabou de sair de um curso técnico, estudou por conta própria, fez um *bootcamp* ou cursou uma universidade e está em seu **primeiro emprego** na área de engenharia de software. Afinal, o mercado é muito variável e temos pessoas com anos de experiência que ainda estão em cargos de júnior, a depender do que a empresa acredita que seja uma pessoa júnior.

Essa pessoa costuma ter um conhecimento teórico ou muito básico sobre computação ou desenvolvimento de software. Não é alguém que deveria assumir a responsabilidade por uma entrega de funcionalidade inteira sozinha, mas fazer pequenas partes para auxiliar essa construção. Pensando em componentes de um sistema, essa pessoa deve ter responsabilidade baixa e pouco impacto na empresa, coisas que não atrasam uma entrega de software. Sua principal função é aprender mais e buscar independência.

Como júnior, nós pedimos muita ajuda de outras pessoas, seja da nossa liderança ou pessoa sênior mais próxima ou de pessoas de outros times que possam compartilhar algo que nos ajude a realizar a tarefa. Na maior parte do tempo, estamos quebrando a cabeça com problemas do nosso ambiente de desenvolvimento em vez de realmente codificar a funcionalidade que nos foi pedida. Afinal de contas, nós somos iniciantes.

Nesse nível, o nosso foco deve ser entender como as coisas realmente funcionam. Quando acessamos uma tarefa, precisamos buscar realmente entender o que foi solicitado, procurar aprender os porquês das decisões técnicas anteriores ou mesmo por qual motivo estamos indo para uma direção. É importante absorver o máximo de conhecimento de quem está na área ou na empresa há mais tempo do que nós.

Tabela de atribuições

Experiência	Dependência	Escopo	Impacto
Nenhuma	Alta	Pequenas tarefas + funcionalidades inteiras com algum auxílio	Seu time

Pleno

A promoção de júnior para pleno pode demorar a acontecer ou ser muito rápida, pois depende da empresa ou de alguma nossa atitude diretamente, como trocar de emprego para algum que reconheça você como pleno ou já possuir muito conhecimento e autonomia. Podemos assumir que alguém neste momento de carreira já atua no mercado há dois ou três anos, já possui independência quando assume uma tarefa e também consegue se responsabilizar por funcionalidades inteiras.

Eu costumo brincar que a pessoa pleno é quem move as tarefas e faz a roda girar de verdade enquanto a pessoa sênior está em reunião. Isso nem sempre é uma verdade, é somente uma brincadeira com o fato de que pessoas pleno normalmente estão 100% focadas em escrever código e entregar mais e mais tarefas, enquanto a pessoa sênior já não consegue mais ficar tão próxima assim de pequenas tarefas ou entregas rápidas. Mas vamos conversar sobre senioridade logo depois.

O impacto do trabalho da pessoa nível pleno é sentido pelo seu próprio time. Enquanto a pessoa júnior precisa de ajuda e não consegue entregar tarefas rapidamente, a pessoa pleno está dando tração nas entregas. Vemos muito mais tarefas finalizadas com o nome dessa pessoa pleno do que da júnior ou da sênior.

Aqui, é importante começarmos a focar em melhorar a qualidade das nossas entregas. Aqueles padrões de projetos que aprendemos no começo da carreira ou que estudamos antes de entrar na área agora começam a fazer mais sentido, afinal, já enfrentamos vários bugs que demoraram para serem resolvidos por falta de qualidade de código.

É muito importante neste momento de nossa carreira não nos desesperarmos pelo próximo cargo. A nossa atribuição vai mudar muito quando formos profissionais sêniores, por isso precisamos aproveitar esse momento para que realmente absorvamos os padrões de projetos, padrões de arquitetura, algoritmos, convenções da indústria e também aprendamos com os nossos erros. Se você

conseguir uma boa mentoria neste momento, será ótimo para a sua carreira como um todo.

Tabela de atribuições

Experiência	Dependência	Escopo	Impacto
2-3 anos	Pouca/baixa	Pequenas tarefas + funcionalidades inteiras + resolução de bugs	Seu time

Sênior

O cargo de sênior é uma grande incógnita em nossa área. Existem discussões na comunidade sobre coisas como:

- Uma pessoa com dois anos de experiência foi contratada como sênior em uma empresa X. Essa pessoa é realmente sênior?
- Uma pessoa tem 25 anos de experiência em uma única empresa em um único time, trabalhando em um único tipo de problema a vida inteira. Essa pessoa é realmente sênior?
- Uma pessoa passou três meses em cada emprego nos últimos cinco anos. Essa pessoa é realmente sênior?

Não existe um consenso de mercado do que define uma pessoa sênior com base em tempo de experiência. Portanto, o exemplo que vou colocar à frente sobre tempo é basicamente para exemplificar que é uma pessoa **com experiência de mercado e/ou conhecimento acadêmico** suficiente para assumir as responsabilidades de uma pessoa sênior.

Vamos assumir que nosso exemplo de pessoa sênior seja uma pessoa com cinco anos ou mais de experiência de mercado. Ela já possui uma base sólida de engenharia de software e computação, suficiente para ter autonomia total de decisões técnicas e liderança esporádica em iniciativas do seu time.

O trabalho da pessoa sênior consiste em resolver problemas que ainda não aconteceram e direcionar o time na construção de uma boa arquitetura de software, para que seja fácil adicionar novas funcionalidades, resolver bugs, monitorar o software, ensinar as pessoas abaixo a construir software com qualidade, assim como contribuir em reuniões assumindo o papel de responder pelo time quando algo pode ou não ser desenvolvido no sistema pelo qual você é responsável.

A pessoa júnior, quando coloca um bug em produção, sente-se extremamente culpada por aquele problema. A pessoa pleno normalmente vai ter um sentimento ruim quando uma funcionalidade pela qual é responsável demorou demais para ser entregue ou precisou de várias alterações antes de ser dada como finalizada. Essas pessoas serão "responsabilizadas" (nós evitamos trabalhar com culpa em empresas com boa cultura de engenharia, porém nós assumimos a responsabilidade pelos nossos erros), mas quem recebe realmente uma cobrança por causa desses problemas é quem está no maior nível de senioridade do time.

Em uma conversa com um dos meus gestores, ele comentou que meu objetivo do ano era que o time começasse a assumir mais riscos, que eu ajudasse minha liderança técnica a construir a confiança no time de que o seu código estava bom suficiente para integrar com o código de produção em qualquer momento do dia. Antes disso, houve outro sênior em meu lugar que já havia estruturado uma boa arquitetura para aquele sistema, criado vários dashboards onde conseguíamos acompanhar o nosso produto no ar, além de deixar a cobertura de testes de unidade e integração em ótimo estado.

Ou seja, o time já deveria ter essa confiança, faltava-lhes pouca coisa. O meu trabalho foi descobrir o que faltava e atacar isso. Neste caso, só precisei dar mais visibilidade para o time sobre a saúde das nossas aplicações. Para isso, eu comecei a compartilhar informações sobre a saúde do sistema em todo início de semana. A autoestima do time subiu e eles começaram a aceitar as apostas que o time de

produtos lhes trazia. Isso fez com que nossas entregas começassem a ser mais rápidas e deu mais liberdade para o time de produto explorar possibilidades para os nossos sistemas.

Perceba que o impacto do trabalho de uma pessoa sênior não necessariamente será entregar código mais rápido do que uma pessoa pleno, ou código mais limpo do que qualquer pessoa na empresa. Está muito mais relacionado com uma mudança de postura, com assumir responsabilidades além do código, do que com pequenas entregas.

Lembre-se sempre: escrever código, as pessoas nível júnior e pleno estão fazendo; você recebeu uma promoção para ir além disso.

Tabela de atribuições

Experiência	Dependência	Escopo	Impacto
~5+ anos	Independente	Funcionalidades inteiras + compartilhar conhecimento + tomadas de decisão	Seu time

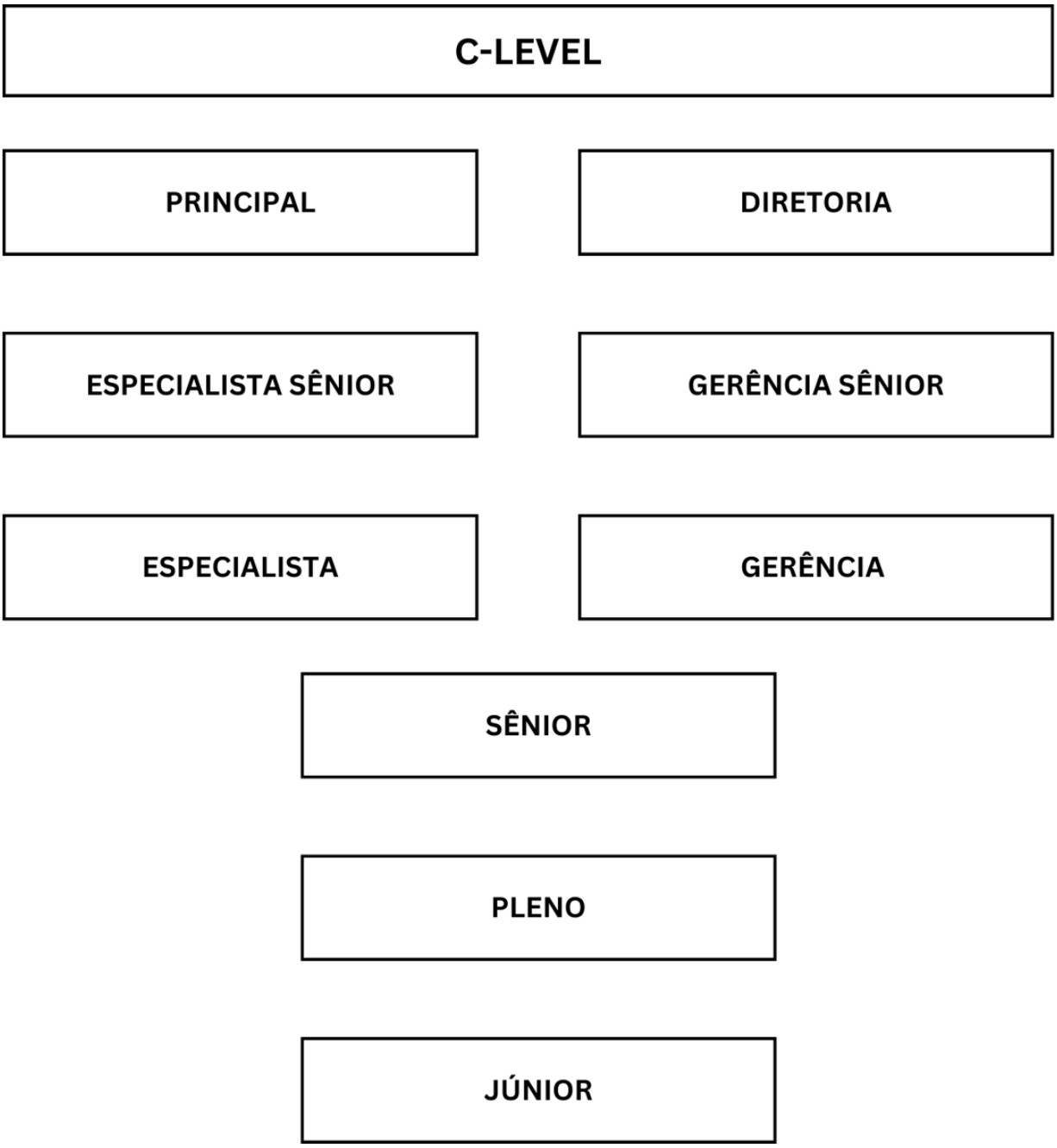
2.3 Carreira em Y

Em nossa trilha profissional, chegará o momento de decidir o que fazer depois de sênior, qual será o escopo do nosso trabalho para os próximos anos depois de conquistar autonomia, independência, liberdade e ter causado muito impacto nas empresas e projetos onde atuamos. Nesse momento, temos a sorte de a carreira não ser mais linear: hoje em dia temos a **carreira em Y**.

Esse termo não é exclusivo de nossa área. Várias profissões hoje em dia possuem o cargo de especialista (ou expert, em algumas organizações). A lógica é que nós trilhamos um caminho reto até aqui e chegamos em um ponto onde podemos continuar aumentando nosso impacto, mas atuando com pessoas ou continuando no caminho técnico.

Aqui eu vou trazer um exemplo de como pode ser a progressão de carreira em Y com base nas empresas onde já trabalhei e também de alguns exemplos que encontramos em sites como o *levels.fyi*. Algumas empresas possuem suas *ladders* (o documento no qual elas organizam seus cargos) abertas ao público, e podemos ainda utilizar isso como base para definir nossa progressão. Meu objetivo nesta parte é esclarecer como funcionaria um possível crescimento em uma empresa com esse modelo de carreira.

Mas, em todos os casos, nós podemos atingir o nível de diretoria de uma empresa, seja pela trilha de gestão ou pelo caminho técnico. Os cargos de especialista e gestão também podem variar, conforme já conversamos, porém seria algo como o da imagem a seguir:



Nessa imagem, temos:

- **Carreira base:** júnior → pleno → sênior
- **Carreira especialista:** especialista → especialista sênior → *principal software engineer*
- **Carreira gestão:** gerente → gerente sênior → diretoria / head de engenharia

Algumas empresas classificam como na tabela a seguir:

Posição	Especialização	Gestão
1 cargo depois de sênior	especialista, especialista I, *staff software engineer*	coordenador(a), gerente
2 cargos depois de sênior	especialista sênior, especialista II, *senior staff software engineer*	gerente, gerente sênior
3 cargos depois de sênior	especialista sênior, especialista III, *principal software engineer*	diretor(a), head de engenharia

Como já vimos, esses títulos podem variar por empresa. Algumas possuem ainda mais cargos antes e depois de principal ou *head* de engenharia. Nós podemos encontrar pessoas técnicas com o título de *senior principal*, *distinguished software engineer* e *fellow* (ou algum outro nome) e pessoas em gestão com títulos como *senior head*, diretor(a) sênior ou vice-presidente.

Por exemplo, **Roberta Arcoverde**, diretora de engenharia no Stack Overflow, comenta que *principal* é o cargo que vem depois de *staff* na organização — isto é, dois degraus depois do sênior. Na empresa, a pessoa no cargo de *principal* terá a responsabilidade por influenciar decisões de tecnologia que afetam a empresa inteira. O nível de influência e responsabilidade aumenta, assim como aumenta a necessidade de conversar e negociar com todos os times, não só com aquele do qual se faz parte. É esta pessoa que define, por exemplo, estratégias pra evolução da arquitetura, escreve PoCs para validar diferentes tecnologias, identifica gargalos de escalabilidade antes que eles aconteçam, propõe mudanças, mas também atua como consultor nos vários times da empresa, não estando alocada em nenhum time de forma fixa.

Laura Madalosso, *principal software engineer* na Dell, comenta que *principal* é uma pessoa sênior com um pouco mais de responsabilidades dentro da organização. Acaba que, como *principal*, ela escreve bem menos código e ajuda seus colegas nas entregas, além de conversar bastante com os designers, product managers e outros stakeholders para entender a necessidade e ajudar o time na entrega, tentando prever cenários futuros.

Talvez algum dia exista um padrão no mercado, talvez isso nunca aconteça. Independente do futuro, agora temos uma opção para continuar atuando como pessoas técnicas nas organizações.

Existem outros modelos de carreira além do Y?

Além da carreira em Y, temos vários outros modelos hoje em dia, como W e X, carreira linear, horizontal e em rede. Pode parecer besteira para algumas pessoas, mas utilizar uma letra como símbolo para organizar as nossas ideias em relação ao direcionamento de carreira é algo interessante, facilita que tenhamos uma visualização daquilo que estamos conversando. Quando pensamos no Y, sabemos que é algo que vai se iniciar em uma linha reta e, em algum momento, acontecerá uma bifurcação em que vamos para um lado ou para o outro, por exemplo. Vamos explorar um pouco os outros modelos.

Um *plano de carreira em W* seria pensado para que a pessoa assuma diretamente a posição de liderança técnica. No caso da área de engenharia, temos o papel de tech lead, que seria um exemplo real de um plano de carreira em W. A pessoa que assume o papel de tech lead é responsável por liderar o time técnico, mas não é responsável por liderar o time de produtos ou é diretamente responsável por novas contratações, feedbacks ou carreira individual dos membros dessa equipe.

Em breve vamos conversar mais sobre o papel e perfil de *tech lead*, mas por enquanto podemos pensar que é uma pessoa que está no meio do caminho entre a trilha de especialista e gestão. Em algumas empresas há até mesmo um modelo híbrido, onde você tem a necessidade de se tornar tech lead antes de se tornar especialista. Nesse caso, nós seguimos um plano de carreira em W de júnior até tech lead e, depois disso, podemos seguir uma das pontas como um plano de carreira em Y.

O *plano de carreira em X*, também conhecido como exponencial, seria o mais generalista possível. A pessoa tem conhecimento técnico e atua em pautas técnicas em alguns momentos, mas também tem conhecimento em gestão e atua em gestão em outros momentos. Ou seja, é totalmente pensado em uma pessoa que não quer seguir uma carreira de especialista ou gestão, mas sim fazer as duas coisas.

Esse modelo de carreira é muito comum em empresas pequenas, onde a pessoa precisa assumir vários papéis diferentes justamente por falta de recursos. Porém, em empresas maiores, é muito difícil que uma pessoa consiga se tornar especialista e gestora ao mesmo tempo. Conversando sobre este modelo com alguns amigos e amigas, eles/elas acham que não conseguiriam se adaptar a algo assim — mas, no meu grupo de amigos e amigas, somos pessoas ou muito técnicas ou que gostam bastante de gestão.

Apesar de não existir um consenso sobre ele, o modelo de *plano carreira horizontal* seria aquele pautado na total autonomia do indivíduo, onde não existem posições ou níveis na organização e a progressão é baseada na complexidade das tarefas desenvolvidas e das responsabilidades que a pessoa assume. Aquela pessoa que sempre gostou de ser a líder em jogos em grupo com certeza vai assumir a responsabilidade de liderança durante o desenvolvimento dos projetos. O mesmo acontece com aquela pessoa que sempre esteve preocupada com todo mundo do grupo: ela vai assumir uma postura de gestão. As demais pessoas ficariam focadas em realizar as outras tarefas.

A *carreira em linha* é a que todos nós conhecemos, aquela cujas opções são estritas. Você vai subindo de nível de júnior, pleno, sênior, especialista ou gestão, diretoria ou *principal engineer*, mas não tem opção de escolher entre um caminho e outro quando chegar a sênior. Empresas com esse modelo de carreira normalmente também não nos possibilitam dar um passo atrás caso sintamos que pegamos o caminho errado — ainda bem que esse modelo de carreira está cada vez mais raro de ser encontrado.

Deixei para falar do *modelo em rede* por último porque, entre os modelos de plano de carreira, esse é o mais flexível de todos. Ele possui as possibilidades que os outros têm, e existem ainda mais opções de cargos que podem ser alcançados dentro da organização. Existe também a possibilidade de migrar de gestão para especialista, e vice-versa, o que o torna um modelo interessante para pessoas que têm um perfil inquieto dentro da organização, as quais, em um momento de suas carreiras, estão muito preocupadas com pessoas e, em outro, estão muito preocupadas com código.

Isso não é um sentimento incomum: eu mesmo passo por isso o tempo todo em meu dia a dia profissional. Em alguns momentos, estou muito preocupado com o futuro profissional das pessoas com quem eu trabalho; em outros, eu participo de discussões para contratação devido à necessidade de aumento do quadro de funcionários para dar conta das futuras demandas; em outros, eu só quero pensar em código e nada mais. E isso é normal, não tem nada de errado com essa flexibilidade de preocupações.

As escolhas não são definitivas

Meu maior medo em seguir por uma carreira de gestão era: se eu já acredito que não gostaria de seguir por este caminho, o que fazer se eu for obrigado a me tornar gestor e ter que trabalhar nessa posição pelo resto da vida?

Na verdade, não funciona bem assim. Com a flexibilidade de carreira que temos hoje em dia, podemos nos mover para a trilha de

especialista ou gestão e fazer o movimento contrário caso percebamos que seríamos melhores na outra ponta do Y.

Você não vai querer fazer esse movimento muitas vezes em sua carreira, porém isso é possível. Vai depender muito mais do lugar onde trabalha ou da sua pretensão salarial, caso tenha que trocar de emprego para fazer essa transição.

Em uma empresa onde trabalhei, tive a oportunidade de entrar em reuniões com meu gestor para que ele me mostrasse como seria o dia a dia atuando naquele lado. Eu pude ver de perto quais eram as dores que ele enfrentava e quais os problemas que estava tentando resolver. Mas, assim como eu fiz esse teste em gestão, eu vi pessoas irem para a trilha de especialista e descobrirem que seriam mais produtivas lidando com pessoas.

Infelizmente, não tem como saber **exatamente** como será o trabalho do próximo cargo até que assumamos essa posição de vez. E pode ser que nós não gostemos do que encontramos. Isso é extremamente comum e, se já aconteceu com você, saiba que muitas outras pessoas já passaram ou estão passando por isso agora.

Por exemplo, **Bruno Rocha**, *principal software engineer* na Red Hat e Fellow na Python Software Foundation, comenta que, em um determinado momento de sua carreira, ele se arriscou para o caminho da gestão de pessoas e o resultado foi que ele sentiu que aquele caminho não era para ele, pois ele sentia muita falta da parte mais técnica da área de engenharia de software.

É bom termos em mente as perspectivas e visões para o futuro de onde estamos no momento. Na empresa atual, seria possível fazer essa movimentação e voltar atrás caso necessário? Se sim, ótimo. Se não, o que podemos fazer?

Como nos tornamos especialistas

Se você se interessou pelo papel de especialista, agora deve estar pensando: como eu faço para trabalhar como especialista? É através de uma promoção interna? Existem vagas no mercado para pessoas especialistas? Essas vagas são mais raras que as de gestão ou mesmo de sênior?

Tudo vai depender de onde você trabalha atualmente ou em que momento de sua carreira você se encontra. Nós podemos assumir o papel de especialista através de uma promoção interna, ou podemos ser contratados para assumir essa posição em outra empresa através das vagas abertas.

Normalmente não encontramos vagas abertas para este nível em empresas pequenas. Acontece muito mais em empresas com um grande time e com muita demanda. Nós podemos encontrar vagas para este nível com o título de especialista (em empresas tradicionais) ou *staff software engineer* (em empresas mais *hipsters* e startups). Mas é importante pesquisar como o mercado está descrevendo estas vagas, pois os títulos podem mudar, as empresas podem chegar a um nome em comum ou mesmo nunca terem um padrão.

Uma busca no Google Jobs no momento em que eu escrevia este livro retornou somente 47 resultados para a entrada "vagas especialista engenharia de software brasil". Quando especificamos melhor a região, aparecem mais opções, como quando procuramos por "vagas especialista engenharia de software são paulo", que retornou 77 resultados, enquanto pesquisar por "vagas staff software engineer brasil" retornou 31 resultados e "vagas staff software engineer são paulo" retornou 24 resultados. Ou seja, para encontrar uma vaga como especialista, precisamos procurar e aplicar para estes dois títulos diferentes (e talvez alguns outros).

Utilizei o Google Jobs porque conseguimos alcançar vagas não somente do LinkedIn, que é o lugar mais utilizado para postar oportunidades em engenharia, mas em outros sites de vagas de emprego. Em todo caso, existem muito mais vagas abertas para

níveis mais baixos. É como acontece com vagas de gestão. Existem muito mais vagas para gerente do que para diretor, e muito mais vagas para diretor do que para vice-presidente. Isso acontece por conta da pirâmide organizacional, onde temos mais pessoas em níveis mais baixos e menos pessoas em níveis mais altos. Normalmente temos apenas um CTO, mas podemos ter várias pessoas engenheiras de software, por exemplo. Entenderemos mais sobre isso nos próximos capítulos, onde vou comentar mais sobre o papel da pessoa especialista, mas normalmente temos uma pessoa especialista por squad, uma pessoa especialista sênior por unidade de negócios e uma pessoa *principal* por unidade de negócios ou para a empresa inteira. Tudo depende do tamanho da empresa e da demanda pelos seus produtos.

No caso de empresas com este cargo, habitualmente temos uma documentação explicando o que faz cada papel e o que é esperado de nós para podermos receber uma promoção. É bem raro que uma pessoa em nível de pleno ou com pouca experiência de trabalho assuma, por promoção, uma posição de especialista em uma empresa, ou mesmo consiga passar em entrevistas de emprego para este nível (mas não é impossível). Mas também será difícil que você receba uma promoção de sênior para especialista caso a sua empresa não tenha esse modelo de plano de carreira. Em alguns casos, a empresa ainda possui cargos de sucessão, ou seja, você precisa esperar que a pessoa que está acima na hierarquia suba mais um nível ou saia da empresa para concorrer à vaga dela (aconteceria o mesmo para pessoas que seguem a trilha de gestão).

O mais comum é que as empresas promovam as pessoas para os próximos níveis. Então começamos no nível mais baixo, que seria em um estágio ou como auxiliar, e vamos alcançando posições até o momento em que escolhemos seguir como especialistas ou gestão, e a empresa quer que tenhamos essa experiência interna de evolução para que fiquemos mais tempo trabalhando com eles. Afinal, com o "tempo de casa", não somente adquirimos conhecimento técnico, mas também sobre o modelo de negócio e problemas específicos da

empresa, e como resolvê-los ou conviver com eles. Isso é muito valioso para a empresa e para nós.

No entanto, isso não é uma regra: existem empresas cuja rotatividade é tão alta que não dá tempo de promover as pessoas para este nível ou simplesmente de conseguir construir um plano sólido para todos de maneira que seja possível manter o equilíbrio entre novos funcionários e promoções. São muitas variáveis. Por exemplo, um profissional que entra com um salário muito alto em uma empresa e depois busca a promoção para especialista pode não conseguir um aumento satisfatório. Nesse caso, contratar alguém de fora facilita uma negociação. Em casos em que entramos com um salário muito baixo, podemos receber uma promoção de cargo, mas, mesmo com um aumento expressivo, ficar com o salário abaixo do mercado, e isso ser um pivô para sairmos e a rotatividade subir. Em outros casos, a empresa possui tantas pessoas com muito tempo de experiência somente trabalhando com ela que precisa contratar alguém de fora no nível de especialista+ para trazer novas ideias e novas práticas para o time.

Em todo caso, existem várias formas de conseguir atuar como especialista, desde uma promoção interna até mesmo tentando uma vaga em outra empresa. O importante é que você saiba que isso existe e que você pode buscar isso, se for o seu objetivo.

Conclusão

Conforme dito anteriormente, não existe uma convenção ou um padrão na indústria sobre quais são nossas atribuições, responsabilidades ou como acontece o pulo para o próximo nível; por isso, é importante que saibamos que essas opções existem. Com as informações deste capítulo, nós já temos os argumentos e exemplos para repassar com a nossa liderança um possível caminho a seguir em nosso plano de carreira.

Em todos os casos, temos que nos manter flexíveis, pois isso depende muito de fatores externos dos quais não temos controle,

como o tamanho da empresa, modelo organizacional, políticas de promoção e muitas outras variáveis.

Também é importante levar deste capítulo que nós temos a opção de trocar de caminho caso trabalhar como especialista ou em gestão não seja o que nos deixa felizes e mais produtivos(as).

Nos próximos capítulos, vamos nos aprofundar mais em detalhes da carreira técnica.

Referências

- *Career Ladders - A sample of career ladders I use for my organization, open sourced for anyone* (DRESSNER, c2023): https://bit.ly/carrer-ladders-dev

- Engineeringladders.com - *A framework for Engineering Managers* (FIORANELLI, c2022) : https://bit.ly/engineeringladders-com

- *Career Ladder* (BETTER, c2023): https://bit.ly/better-engineering-career-ladder

- Levels.fyi: https://www.levels.fyi/

- *What's The Difference Between a Junior, Mid-Level, and Senior Developer?* (ZAWADZKI, 2021): https://bit.ly/honeypot-junior-mid-level-and-senior-developer

- *Y career: understand what it is and its possibilities for professionals* (HR CONSULTANT UK, c2021): https://bit.ly/hrconsultantuk-y-career

- *Plano de carreira em Y: o que é, vantagens e mais sobre esse modelo* (FERRACCIU, 2021): https://bit.ly/gupy-carreira-em-y

- *Tipos de plano de carreira* (INDEEED, 2023): https://bit.ly/indeed-carreira

- *Trilhas de carreira não lineares*: o desenvolvimento profissional (BRITO, 2021): https://bit.ly/espm-trilhas-nao-lineares

Capítulo 3
O papel de especialista

Vamos imaginar que alcançamos a senioridade e que estamos em um caminho tradicional de carreira. Finalmente chegamos no momento em que será necessário tomar a decisão sobre qual seria o nosso próximo passo na área de engenharia de software e, depois de entender a evolução profissional, e realmente nos entendermos como profissionais, percebemos que não queremos seguir a trilha para gestão e, então, escolhemos seguir para a trilha de especialista.

Entretanto, mesmo escolhendo onde desejamos investir nossa energia e foco para os próximos anos, ainda está tudo muito nebuloso... Qual é o real papel de uma pessoa especialista? Onde essa pessoa se encaixa em um time? Onde essa pessoa se encaixa em uma empresa?

São muitas perguntas sem respostas dentro das empresas e até mesmo sem uma definição formal na comunidade de software. Pensando em clarear nossa mente, vamos nos aprofundar em cada detalhe do papel de uma pessoa especialista neste capítulo — lembrando que são sugestões catalogadas a partir das minhas pesquisas e experiências profissionais.

Mas, antes de responder a essas perguntas, precisamos analisar o mercado como um todo e discutir questões como **escopo de trabalho** e a **falta de padrão** no mercado de tecnologia, para depois conseguirmos realmente aprofundar no papel de especialista sem acreditar que teremos uma única maneira de fazer as coisas.

3.1 Diferenças de escopo e cargos no mercado de tecnologia

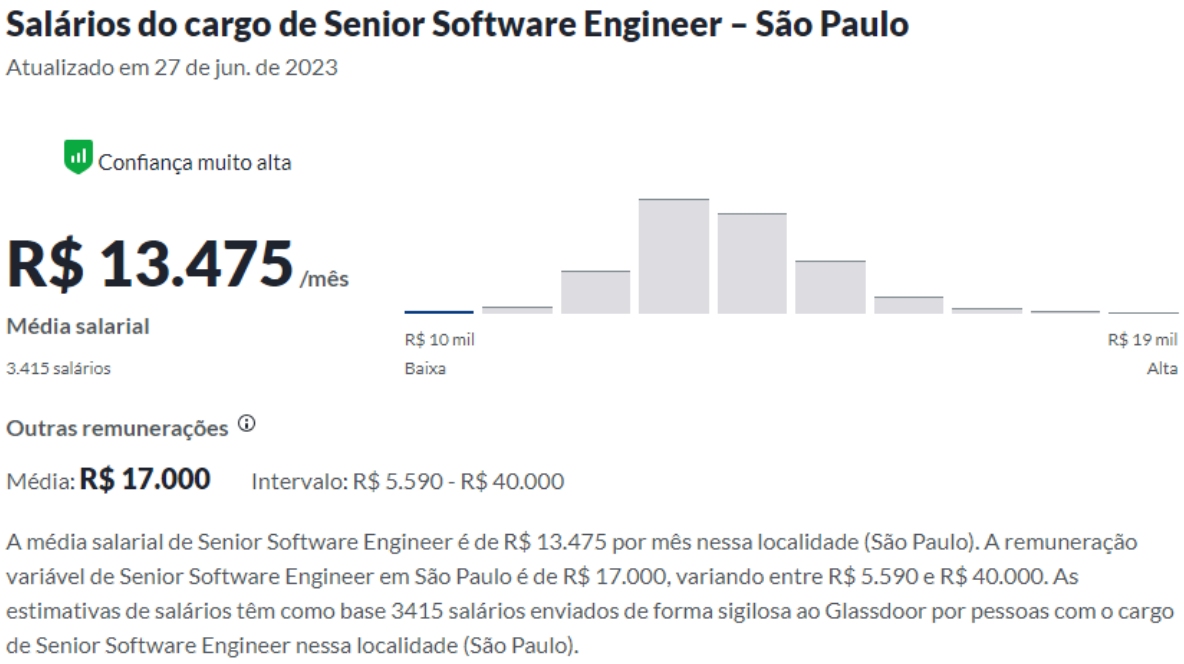
Parte do nosso trabalho como pessoas desenvolvedoras de software é descobrir qual é o nosso papel dentro da organização em que estamos. Pode soar estranho pensar que nós assinamos um contrato para trabalhar como júnior, pleno, sênior ou especialista+ e mesmo assim não temos certeza do que vamos fazer dentro de uma organização. Isso acontece porque o nosso trabalho, apesar de estar dentro de uma área de ciência exata, também é criativo e proativo. Isso não é exclusividade da nossa área, outras várias profissões conhecidas como "altamente qualificadas" não têm escopo fechado dentro das organizações.

Trabalhar com escopo fechado seria como no exemplo de uma linha de produção, onde estamos posicionados(as) em uma parte específica da construção de algo e nosso papel é realizar somente uma tarefa durante o dia inteiro. Imagine que, na construção de um carro, nosso trabalho é receber a carcaça do que será um produto completo um dia e colocar os faróis no lugar. Nosso escopo está definido como colocar faróis. Talvez isso mude em algum momento, dependendo da necessidade da linha de produção, porém, ao trocar de posição na linha, vamos ter novamente um escopo fechado do que fazer naquela etapa.

O trabalho com engenharia de software possui escopo aberto e amplo. Em um momento estamos escrevendo código; em outro momento, estamos realizando uma reunião discutindo a criação de novas funcionalidades dentro de um software; em outro, estamos resolvendo um bug crítico em produção, e assim vai. Mas, na maioria das vezes, por mais simples que seja a tarefa, não existe só um jeito de fazer, e precisamos decidir qual será o modo como a faremos. Pensando em um exemplo mais simples possível, a criação de um novo módulo dentro da nossa aplicação para fazer login pode significar a escrita rápida de um código que se conecta com o servidor ou a construção de algo mais robusto, que pode ser utilizado em vários projetos ou pela empresa inteira (dependendo do que for, até mesmo pela indústria inteira).

Inclusive, pensando em cargos, muitas vezes acontece de estarmos atuando em um nível na empresa atual e recebermos avaliação abaixo desse cargo em uma entrevista de emprego ou quando trocamos de time dentro da mesma empresa. Isso porque o escopo dentro daquela outra empresa, que nos avaliou para baixo, é mais amplo do que o local onde estamos atualmente. Não tem nada de errado conosco, é somente a falta de padrão do mercado. Isso acontece tanto pelo nosso nível técnico quanto pelas experiências que já vivenciamos, assim como por pontos comportamentais, como comunicação, postura de liderança e outras habilidades que ficaram conhecidas como *soft skills*.

Uma busca no Glassdoor (um site de vagas, avaliações e outros dados sobre empregos e empresas) para *senior software engineer* para a cidade de São Paulo nos traz uma disparidade imensa em relação ao salário de pessoas desenvolvedoras.



Percebemos que existem pessoas recebendo 10 mil reais por mês e outras com o salário de 19 mil. Em uma empresa onde trabalhei, o salário de uma pessoa sênior era de 22 mil reais. O de especialista passava dos 25 mil, enquanto em outra empresa o salário de sênior era de 14 mil (teto) e de especialista, 19 mil. O lugar com o maior

salário era uma startup considerada unicórnio (empresas que atingem uma avaliação de mercado de US$ 1 bilhão antes de serem cotadas em bolsa) e o lugar com o salário menor era um banco tradicional e extremamente consolidado.

Essa diferença salarial acontece por diversos fatores, tanto pelo escopo de trabalho da pessoa desenvolvedora, que pode ser mais amplo em uma empresa com o salário tão alto, quanto por questões não diretamente relacionadas a nós profissionais, como o risco de trabalhar em uma startup e ela quebrar antes mesmo do IPO. As startups também costumam contratar pessoas com o salário maior devido a não ter estrutura organizacional suficiente para que o escopo de trabalho seja muito bem definido, precisando assim que as pessoas trabalhem com um nível de independência muito alto, muitas vezes sem nenhuma ajuda da gestão ou liderança direta. Big techs como Google, Mercado Livre e Meta já têm uma estrutura maior, mas também costumam contratar pessoas com altíssimo nível de conhecimento técnico e escopo abrangente, por isso o salário nessas empresas é maior.

IPO (Initial Public Offering, ou oferta pública inicial, em português): Representa a primeira vez que uma empresa receberá novos sócios realizando uma oferta de ações ao mercado. Ela se torna, então, uma companhia de capital aberto com ações negociadas na Bolsa de Valores, podendo ser a Bolsa de Valores brasileira ou estrangeira.

Perceba sempre que: nós podemos atuar como sênior em um lugar onde o escopo é fechado, tendo apoio direto da liderança e gestão realizando algumas tarefas de descoberta e construção de objetivos claros para o nosso trabalho, ou, com o mesmo nível de conhecimento e tempo de experiência, trabalhar em outra empresa na qual precisamos desbravar o que precisamos fazer, assumindo muito mais riscos do que em lugares com estrutura organizacional maior. Mas isso não significa ser melhor ou pior. São escolhas

profissionais baseadas em nosso perfil. Nós também precisamos entender onde nós nos encaixamos melhor dentro deste mercado, pois é o que vai garantir o nosso sucesso profissional.

3.2 O papel de uma pessoa especialista

Durante a minha carreira, eu tive contato com diversas pessoas em posição de especialistas, em vários níveis profissionais e experiências também muito variadas. Trabalhei com pessoas que eram especialistas em uma linguagem de programação, em uma área de atuação, como front-end, back-end, mobile ou embarcados, assim como tive contato com pessoas generalistas (conhecidas como *full-stacks*). O mercado tem espaço e gosta de todos esses perfis.

Independentemente da escolha de especialização que nós fazemos (seja atuar como generalista ou com foco em uma tecnologia/frente de desenvolvimento), o nosso papel estará sempre em sermos uma referência dentro da empresa ou do projeto em que atuamos.

Você pode ter uma pessoa tech lead em seu time, a pessoa que atua diretamente com a liderança **da squad** e direciona o time para realizar as entregas relacionadas a um ou mais projetos desse time. Essa pessoa é vista como a referência técnica para essa equipe. É ela quem direciona a arquitetura, boas práticas, convenções e tudo relacionado à saúde dos softwares ou do sistema do qual o time cuida. Normalmente, tech lead é uma pessoa em nível de especialista, mas em algumas empresas esse papel é intermediário entre gestão de pessoas e tarefas técnicas, enquanto quem está com o cargo de especialista em si não tem responsabilidade de gestão de pessoas. Porém existem lugares onde a pessoa em cargo de gestão atua como tech lead esporadicamente e os times só contam com a pessoa sênior.

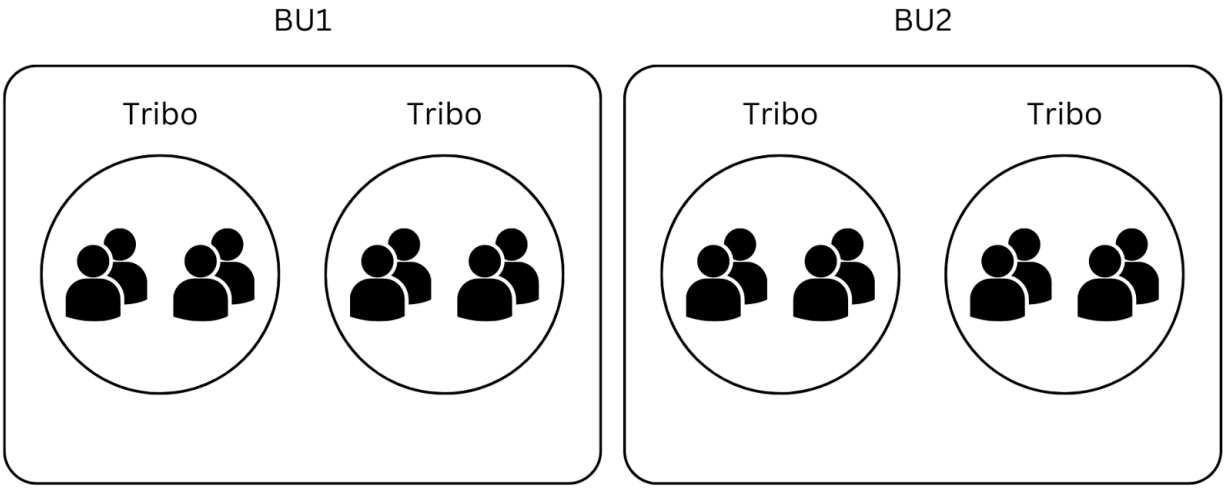
Inclusive, em algumas empresas, tech lead é um chapéu, que passa entre pessoas nível pleno e sênior e vai sendo trocado por pessoas do mesmo time. Mas, na grande maioria das vezes, a pessoa tech lead nada mais é do que alguém que assumiu o **papel** de referência técnica para o time, o que se assemelha muito com o **cargo** de especialista.

Ser referência é algo muito subjetivo, afinal aquela pessoa em nível pleno que conhece tudo sobre uma tecnologia específica pode ser a nossa referência sobre aquele tema em nosso time. Por isso, o papel da pessoa especialista não é somente conhecer muito sobre algo, também está relacionado com compartilhar este conhecimento com o time, com o objetivo de equalizar a equipe, além de ser a pessoa que vai tomar decisões difíceis ou apoiar as pessoas que estiverem em posição de tech lead e gestão nessas tomadas de decisão quando necessário.

O papel de especialista está relacionado à liderança, e isso não é sobre assumir o cargo/chapéu de tech lead, mas direcionar iniciativas e pessoas com um objetivo técnico. Normalmente estamos com o foco em aumentar a qualidade do código, melhorar a observabilidade ou sustentabilidade de um sistema, garantir que os times estão seguindo boas práticas e apoiando as equipes para que tenhamos uma boa experiência de desenvolvimento de software, além de confiança nos sistemas que estão em produção.

Nós temos o papel de escrever/trazer propostas de melhorias que serão discutidas entre especialistas, depois com as lideranças técnicas e, por fim, implementadas pelos times de engenharia com o nosso apoio (seja com a mão na massa ou não).

Normalmente a pessoa especialista não está ligada a um time, mas à empresa. Vamos imaginar uma estrutura organizacional para facilitar o entendimento. Imagine que temos uma empresa com duas unidades de negócio (BU - *Business Unit*), dentro dessas unidades, temos duas tribos e, dentro dessas tribos, dois times (squads).



Agora pense que o time 1 da BU 2 está iniciando um novo projeto. Para que este projeto esteja alinhado com os padrões definidos pela empresa ou pela indústria, é importante que aconteça uma conversa prévia ao início da codificação. Para isso, temos a pessoa especialista (ou as pessoas), que estará ali para auxiliar que o que o time 1 da BU 2 está fazendo não extrapola as convenções que os times 1 e 2 da BU 1 ou o time 2 da BU 2 estão fazendo. Isso vai garantir uma estabilidade organizacional e possibilidade de compartilhamento (seja de código, conhecimento ou qualquer outra facilitação) entre toda a empresa.

Perceba que a pessoa especialista serviu como uma cola entre todos os times. Nós estamos impactando a empresa muito mais pelo fato de estarmos em vários locais tentando organizar e manter as coisas organizadas do que literalmente escrevendo algum código. E existem várias possibilidades para que façamos isso, seja entrando em uma reunião com os times, seja escrevendo documentações para que todos sigam estes padrões ou automatizando tarefas.

Uma vez me foi atribuída a função de melhorar a qualidade dos softwares do sistema pelo qual uma squad era responsável. A empresa não possuía uma documentação sobre o que era essa qualidade, por isso foi necessário que eu primeiro buscasse entender o que era a qualidade que eles buscavam. Era deixar o código mais legível? Era deixar a arquitetura escalável? Era trabalhar em

observabilidade? Diminuir a quantidade de bugs? Existem muitos tópicos dentro de qualidade de software.

Depois de entender o que eles precisavam, escrevi uma documentação com parâmetros de qualidade dentro dos nossos sistemas. E comecei a entrar nas reuniões de planejamento do time para tentar levantar débitos técnicos que poderiam ser resolvidos pensando em seguir aquela documentação.

Em seguida, eu comecei a mexer no código e adicionar automações para que não fosse mais possível diminuir a quantidade de testes nem subir código sem revisão de pares, bem como adicionei outros meios de atingir a qualidade que estava naquela documentação. Finalmente, depois de oito meses desse trabalho, chegamos aos parâmetros de qualidade que a gestão havia me pedido.

Em alguns lugares onde trabalhei, as pessoas especialistas desenvolveram templates para que os times utilizassem na hora de iniciar um novo projeto, fosse um *boilerplate* para o Spring Boot, um módulo para utilizar no Express ou um repositório que servia como base na hora de inicializar outro repositório, além de escreverem documentações extensas sobre qual era o padrão de trabalho na empresa. Tudo isso está relacionado com tentar organizar a casa.

Mesmo sem um padrão de mercado definido, percebemos que o trabalho da pessoa especialista está muito relacionado com o direcionamento técnico dentro das empresas, dos projetos e com liderança esporádica. É necessário que descubramos como nossas habilidades podem ser mais bem aproveitadas dentro da organização.

3.3 Níveis de especialistas

Assim como temos os níveis júnior, pleno e sênior antes do cargo de especialista, quando assumimos esta trilha, abrem-se novos cargos e

atribuições diferentes também baseados em nosso tempo de experiência, na atuação e na empresa onde trabalhamos. É como acontece com a pessoa que escolhe o cargo de gestão, ela terá o nível *engineer manager*, *senior engineer manager*, *head* e por aí vai.

Para facilitar o entendimento, eu escolhi os títulos utilizados internacionalmente: *staff software engineer*, *senior staff software engineer* e *principal software engineer*. Mas sua empresa pode ter algo como: especialista I, especialista II, especialista III ou algo totalmente diferente.

Esses cargos, conforme já conversamos, estão também relacionados com liderança técnica, portanto, uma pessoa em cargo de *staff* está (em nível organizacional) alinhada ao nível de gestão; a *senior staff*, ao nível de gestão sênior; e a *principal*, ao nível de diretoria/head de engenharia. Isso também pode variar. Em alguns casos, a pessoa em nível *principal* está abaixo da diretoria de engenharia e teria um cargo a mais, o *senior principal*, que seria quem realmente está alinhado à diretoria.

É importante entendermos isso porque percebemos a mudança do impacto do nosso trabalho comparando os cargos de gestão e especialista lado a lado. Até o nível de sênior, nosso trabalho está totalmente atrelado ao nosso time. A pessoa sênior também realiza ações como as de especialista, como liderar uma iniciativa, direcionar o time, compartilhar conhecimento, porém sempre impactando somente este time. A pessoa especialista passa a impactar vários times.

Vamos lembrar da organização tradicional da empresa, pensando em uma pirâmide organizacional e utilizando a nomenclatura que a maioria utiliza:

- um **time** é um grupo de pessoas com uma missão;
- uma **tribo** é um grupo de vários times com uma missão em comum e algumas missões individuais;

- uma **unidade de negócios** são várias tribos com uma missão em comum;
- uma **empresa** é um aglomerado de todos os outros com o objetivo de gerar lucro.

Portanto, temos o seguinte racional para a pessoa especialista:

Cargo	Experiência	Dependência	Escopo	Im
Staff	8+ anos	Independente	Direcionamento técnico de múltiplos times	Trib
Senior staff	Indefinido	Independente	Direcionamento técnico de múltiplos times de especialistas	Uni de neg
Principal	Indefinido	Independente	Direcionamento técnico da organização como um todo	Em

Uma pessoa *staff* normalmente tem responsabilidade com uma tribo. Ela pode ser responsável por mais tribos, dependendo da empresa, porém o mais correto é que, nesse primeiro nível de especialista, ela tenha responsabilidade com um grupo de times. Conforme sobe de nível, a pessoa *senior staff* tem a responsabilidade de cuidar das outras pessoas *staffs*, por isso acaba impactando uma unidade de negócios inteira. No "último" nível da hierarquia de especialistas, a pessoa *principal* é responsável por todas as pessoas especialistas, o que impacta a empresa inteira.

Algumas empresas possuem organizações ainda mais quebradas e especialistas ainda mais independentes, por isso ainda existem cargos abaixo ou acima de *principal software engineer*. Algo interessante de se refletir é que, enquanto uma pessoa em nível de diretoria da empresa está pensando na produtividade da empresa como um todo (lucro, custo operacional, modelo de negócios etc.), a pessoa *principal* está ao lado dela pensando em coisas similares e outras mais relacionadas aos sistemas da empresa e em como esses sistemas se encaixam no modelo de negócio e com preocupações de negócios.

Para exemplificar, em um local onde trabalhei, passei pelo desafio de ser responsável pela organização do trabalho nas aplicações web de uma tribo nova. Parte do meu trabalho era fazer com que todos da tribo seguissem os padrões da empresa para o desenvolvimento front-end web, pois possuíamos templates para iniciar os projetos, bibliotecas homologadas que deveriam ser usadas e convenções de qualidade que deveriam ser seguidas. Em determinado momento eu precisei tomar uma decisão técnica que impactava não somente essa tribo, mas toda a unidade de negócios onde eu estava. Como eu estava extrapolando o meu escopo, precisei alinhar com o especialista acima de mim e com o *principal software engineer* para que, caso a minha alteração acontecesse, o restante da empresa também passasse a seguir aquele padrão.

A aprovação dessa alteração dependia do *principal*, porque ele tem o contexto da empresa como um todo. Talvez outros times tivessem os mesmos problemas que eu fui resolver ou mesmo a empresa já estivesse ciente destes problemas e já estava se movimentando para solucionar essas questões. A única maneira de saber seria conversar com cada time da empresa ou conversar com quem direcionava a parte técnica, o *principal*.

Ainda existe muito a se desbravar e organizar no nosso mercado para que estes cargos tenham sentido em algumas organizações ou mesmo para que nós realmente saibamos o que precisamos fazer no nosso dia a dia e não acabemos trabalhando com acúmulo de funções, recebendo pouco por isso. Mas, dentro do possível, serão estes os níveis que encontraremos nas empresas.

O caminho para subir de sênior para *staff*, de *staff* para *senior staff*, de *senior staff* para *principal* acontece como comentei no capítulo anterior: pode ser através de uma promoção interna ou concorrer a uma vaga em outra empresa. Também vai acontecer o caso de empresas nas quais o próximo nível é um cargo de sucessão, e você teria que aguardar a posição ser liberada por alguém para poder concorrer.

Em meu caso, as duas vezes (até agora) em que assumi o papel de especialista foi através de um convite para começar a realizar as tarefas dessa posição e descobrir se eu realmente gostaria de seguir nessa direção. O que também é muito legal de se fazer, uma vez que, enquanto não temos o cargo "cravado" em nosso crachá, não sentimos tanta pressão ou medo de cometer erros. No entanto, eu trabalhei em um lugar onde a promoção era por sucessão e eu preferi concorrer a vagas de especialista no mercado de trabalho.

3.4 Cargos além de principal

Principal pode ser o último cargo na maioria das empresas. Porém, em empresas gigantes e, principalmente, internacionais, cujos times são distribuídos em vários países, ainda existem cargos além de *principal* para tentar manter a ordem e a organização dentro desse grupo tão grande.

É comum encontrarmos, por exemplo, o cargo de *senior principal*, que seria a pessoa responsável por um grupo de *principals*. Novamente, dentro do desenho organizacional de uma empresa, temos aquele modelo de pirâmide, onde alguém acima direciona um grupo de pessoas abaixo. A pessoa *senior principal* entra como alguém que tem não somente a capacidade de organizar a empresa, mas de lidar com problemas de múltiplas regiões e múltiplos negócios. Pode acontecer de este cargo existir em empresas não internacionais também, mas sempre vai existir em empresas **muito** grandes.

Apesar de serem raros no Brasil, ainda temos dois outros cargos que podem vir depois de *senior principal software engineer* ou em paralelo, dependendo de como a organização se posiciona. São os cargos *distinguished software engineer* e *fellow software engineer* (ou apenas *fellow* em algumas empresas), que desempenham um

papel além do organizacional, liderando pesquisas e o desenvolvimento de novas tecnologias e soluções para a empresa.

Como exemplo para esses níveis, podemos citar a *ladder* da GitLab, que é aberta ao público e open source em sua página de carreiras (link no final do capítulo). Neste documento, vemos que uma pessoa em nível de *distinguished*, assim como *principal*, está para a organização no mesmo nível hierárquico de uma pessoa em nível de diretoria. A tabela a seguir foi extraída do site de carreiras da GitLab.

Management	**Individual Contributor**
Senior Director	Senior Distinguished
Director	Distinguished / Senior Principal / Product Manager

Para essas posições, teremos atribuições de impacto não somente em nossa empresa, mas em uma indústria como um todo. A pessoa em cargo de *distinguished software engineer* teria a responsabilidade de atuar em um nível mais alto de resolução de problemas do que a pessoa que está como *principal software engineer*, pois, enquanto a pessoa *principal* precisa cuidar e direcionar a empresa, a pessoa em nível de *distinguished* está movimentando a comunidade de software e indústria buscando a criação de novas tecnologias ou soluções para problemas que talvez ainda nem foram catalogados.

Em nível de *fellow software engineer*, a pessoa possui uma liberdade de atuação ainda maior do que em todos os níveis anteriores e seu impacto pode ser visto não somente na nossa indústria, mas em várias áreas da sociedade. Assim como uma pessoa *distinguished software engineer*, a pessoa em nível de *fellow* também atua em pesquisa e desenvolvimento, porém com uma liberdade de escolha ainda maior sobre onde investir o seu tempo. É alguém com muita experiência e muitas pesquisas e projetos de sucesso em seu

currículo, o que faz com que a empresa confie ainda mais em seu julgamento e decisões, deixando ao seu cargo o que ela vai fazer e como vai fazer. Normalmente a pessoa *distinguished software engineer* é promovida a *fellow software engineer*, mas também pode acontecer de uma pessoa em nível de *principal* ou *senior principal* ser promovida a *fellow*, pois vai depender da capacidade e dos projetos/pesquisas que essa pessoa já realizou.

É comum ver pessoas em nível *distinguished* e *fellow* criando novas linguagens de programação, inventando novos produtos de cloud (como bancos de dados, servidores etc.), criando novas tecnologias de inteligência artificial, entre outras coisas. É um nível de impacto muito grande e, raramente, é uma demanda somente da empresa. Essas pessoas têm uma grande conexão com a comunidade de software, universidades e institutos de pesquisas.

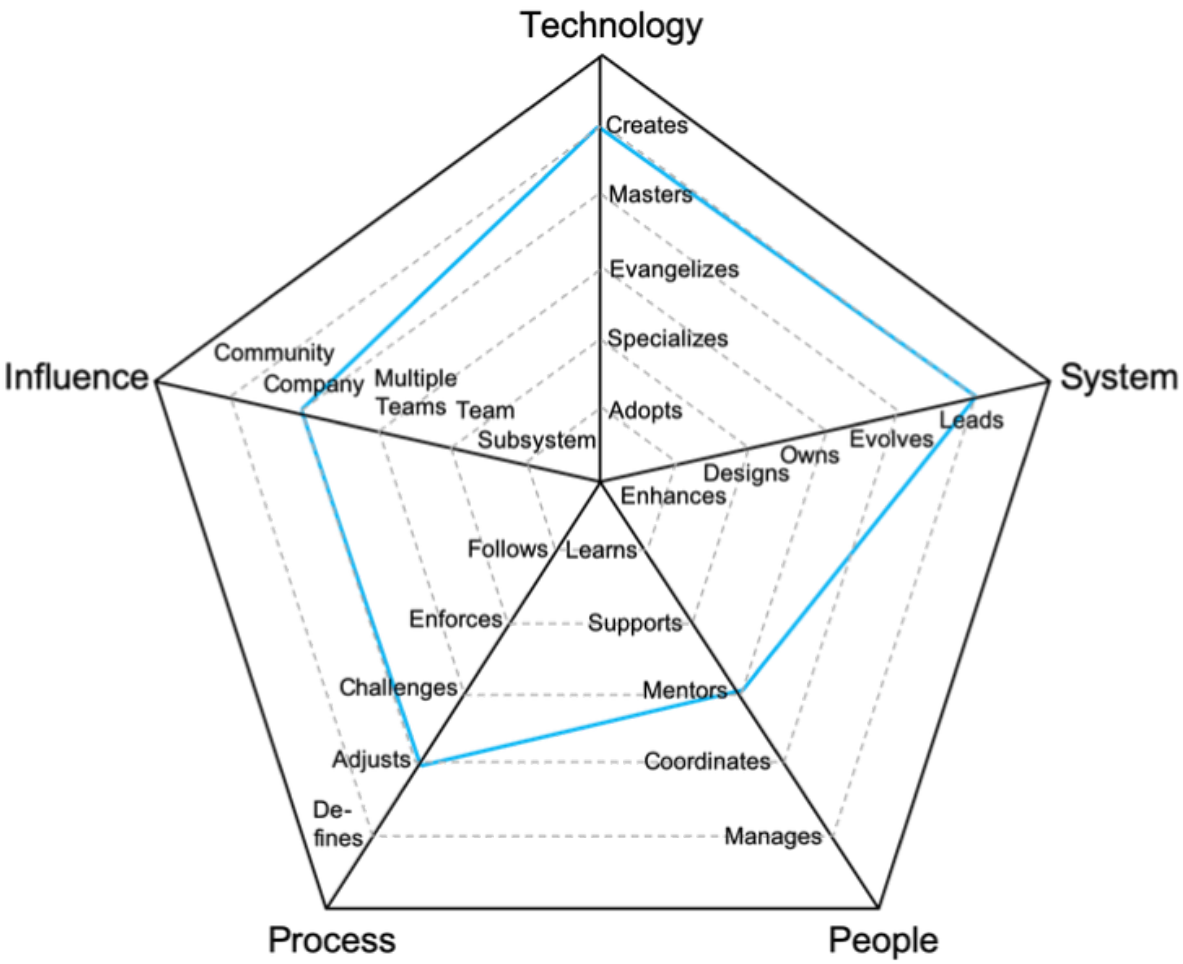
Pode acontecer de vermos o cargo de *distinguished*, mas não o de fellow, ou o de *fellow* e não o de *distinguished*, pois isso também depende de como a empresa se organiza e do que ela precisa. Normalmente pessoas em nível de *distinguished* estão trabalhando em pesquisa em conjunto com pesquisadores, mas podem ou não estarem liderando essas iniciativas e/ou serem pesquisadoras de carreira.

Esses cargos são mais comuns em empresas como Google, Microsoft, Meta, Amazon, Mercado Livre, ThoughtWorks, GitLab, ZUP Innovation e outras empresas que, além de possuírem grande domínio de negócio, também criam novas tecnologias e trazem inovações para toda a indústria. As pessoas destes níveis são as que convencem a empresa sobre onde e como devem investir, e lideram essas iniciativas.

De todo modo, vale sempre lembrar que, dependendo da nossa atuação profissional, nós conseguimos impactar a indústria em diversos níveis profissionais em nossas contribuições individuais em comunidades de software e projetos open source. A grande diferença é que nós fazemos isso como um trabalho paralelo,

enquanto uma pessoa nível staff+ possui essa atribuição em sua *career ladder* dentro das empresas — e isso conta muito em sua promoção, pois parte da atribuição de uma pessoa nível staff+ é justamente contribuir com a comunidade de software e indústria.

No site *engineeringladders.com*, que é outra ótima referência sobre níveis profissionais em engenharia de software, nós temos uma definição similar. Veja:

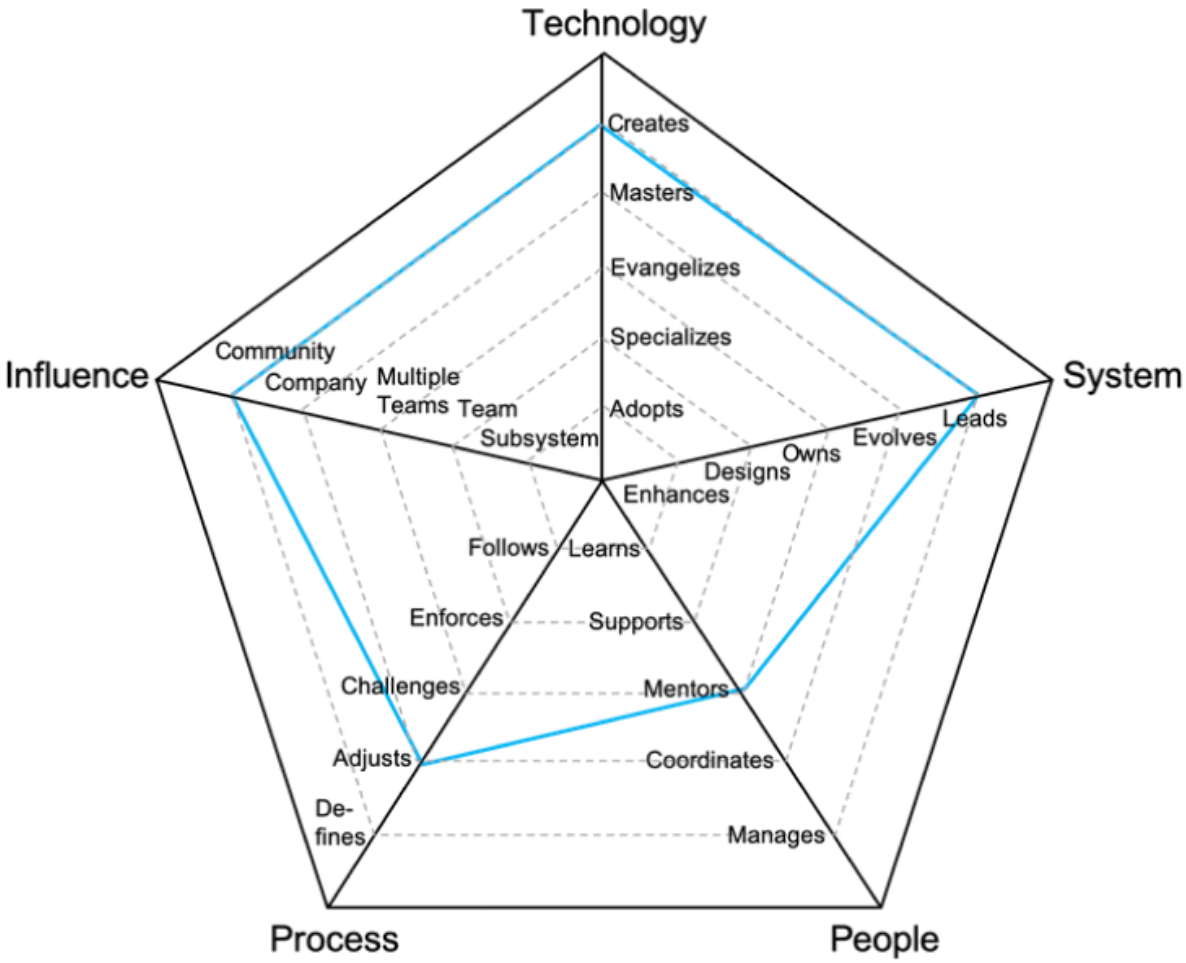


Aqui temos um exemplo de definição de atribuição de uma pessoa nível *principal*, a qual o site organiza como:

- Criação: projeta e cria novas tecnologias que são amplamente utilizadas por equipes internas ou externas;

- Liderança: lidera a excelência técnica do sistema e cria planos para mitigar indisponibilidades;
- Mentoria: mentora outras pessoas para acelerar o crescimento de suas carreiras e as encoraja a participar;
- Ajustes: ajusta os processos da equipe, ouvindo feedback e orientando-a nas mudanças;
- Empresa: causa impacto em toda a organização de tecnologia.

Na figura a seguir, temos um exemplo das atribuições de uma pessoa em nível *distinguished* e *fellow*;



E o site organiza as atribuições desses níveis profissionais como:

- Criação: projeta e cria novas tecnologias que são amplamente utilizadas por equipes internas ou externas;

- Liderança: lidera a excelência técnica do sistema e cria planos para mitigar indisponibilidades;
- Mentoria: mentora outras pessoas para acelerar o crescimento de suas carreiras e as encoraja a participar;
- Ajustes: ajusta os processos da equipe, ouvindo feedback e orientando-a nas mudanças;
- Comunidade: causa impacto na comunidade de tecnologia.

Causar impacto na comunidade de tecnologia impacta, por consequência, outros setores da sociedade. Uma pessoa que cria uma nova linguagem de programação que vai resolver problemas específicos de software que até então não tinham solução vai impactar toda a indústria e a sociedade.

3.5 Onde a pessoa especialista se encaixa na estratégia de uma empresa

Entender onde alguém em posição de especialista se encaixa dentro de uma empresa é bem complicado, uma vez que cada empresa pode ou não precisar deste profissional, o que vai depender do tamanho da organização ou até mesmo do tipo de problema que ela resolve em seu dia a dia. Na maioria das vezes, empresas pequenas não conseguem contratar um profissional tão sênior, não só pelo salário, mas também por não possuir desafios suficientes para justificar essa contratação. É uma questão de retorno do investimento. Não é todo projeto que precisa de especialistas; nem todo time na organização precisa de todos os perfis (júnior, pleno, sênior ou mais). Por isso, é importante que entendamos onde se encaixam as pessoas especialistas, pois a falta do próximo cargo pode ser algo que nos afete pessoal, profissional e financeiramente caso estejamos procurando atuar neste nível em uma empresa que não precisa disso.

Talvez você esteja atuando como sênior em uma empresa já há alguns anos e nem sinal de uma promoção para especialista, mas será que o lugar onde você está precisa de alguém nessa função?

É importante que percebamos que esse cargo existe para resolver problemas organizacionais que não acontecem em todos os estágios de um negócio. Dependendo do tamanho da empresa, a liderança técnica consegue ser suficiente para direcionar o time e se comunicar com outros times e com outras unidades, não sendo necessário alguém com outra atribuição servindo como cola entre outras partes da organização.

Na maioria das vezes, empresas muito pequenas não conseguem contratar nem mesmo um profissional mais sênior para direcionar o time e ajudar as pessoas mais juniores a subirem seu nível técnico. Isso acontece com tanta frequência que encontramos diversos relatos em comunidades, eventos e fóruns sobre pessoas em nível júnior trabalhando sozinhas em projetos de pequeno e médio porte em empresas menores. Isso não só pelo salário, mas também por não ter o desafio que vai motivar essa pessoa a continuar ali. Muitas pessoas da nossa área trocam de empresa porque gostariam de ter mais desafios técnicos do que o que estão enfrentando atualmente. Ou seja, um(a) profissional mais sênior sem um bom salário e sem desafios técnicos provavelmente sairia da empresa em poucos meses.

Normalmente, empresas com um longo tempo de vida e com um bom número de funcionários têm cargos bem definidos. Eu já conversei com muitas pessoas que reclamam de que, em sua empresa, não existe uma clareza do que fazer para ir para o próximo nível. Talvez isso acontece porque as empresas não precisam desse próximo nível. Dificilmente vamos precisar de uma pessoa especialista quando a empresa está em um momento de vida onda a comunicação acontece entre todos sem dificuldade, onde a diretoria técnica (seja alguém em cargo de CTO ou head) tem tempo suficiente para cuidar de negócios, mas também de engenharia de software.

Em um momento da vida trabalhei em uma empresa que estava em crescimento. Quando entrei nessa empresa, o time de engenharia contava com 60 pessoas. Em 4 meses, o time já passava de 300 pessoas. Durante esse crescimento, o antigo CTO deu espaço para outra pessoa e foi atuar em produtos, trabalhando como CPO. Este novo CTO trouxe diversas mudanças para a empresa e muita gente reclamava (eu inclusive) pensando "*por que ele não cuida de outras coisas e deixa a parte mais específica de engenharia conosco?*". Mas isso é muito comum: enquanto ele não tinha pessoas em nível de especialista suficientes para que ele não precisasse acessar um *pull request* e fazer um *code review*, ele mesmo estava quase onipresente em todas as discussões do time de engenharia. Uma hora o real trabalho como CTO tomou conta de todo o seu tempo livre. Porém também existem pessoas nestes cargos que simplesmente gostam de participar de discussões de baixo nível, e elas realmente estarão em todo canto, independente de quanto trabalho possuam lá em cima.

Então, a pessoa especialista, além de precisar de um certo tamanho de empresa para ter desafios (ter trabalho suficiente), também pode ser chave para dar manutenção na empresa que acabou de crescer e vai precisar de reestruturações e novas convenções.

Normalmente nós atuamos apoiando a gestão, a diretoria ou C-levels a manter a empresa produtiva. Enquanto meus gestores cuidavam da carreira das pessoas dos meus times, eu cuidava dos sistemas que estavam no ar. Ele não precisava se preocupar com bugs, logs ou qualquer coisa relacionada ao crescimento dos sistemas, somente com a parte de pessoas do dia a dia na empresa.

Conclusão

Mesmo que não tenhamos os padrões muito bem definidos, conseguimos traçar similaridades entre as empresas, e eu tentei trazer as experiências pessoais de várias empresas onde trabalhei para deixar mais claro o papel de uma pessoa especialista dentro de uma organização.

Para facilitar nosso entendimento, vou trazer a tabela que utilizamos no capítulo anterior com os novos cargos que vimos aqui, para que analisemos antes de seguir em frente.

Cargo	Experiência	Dependência	Escopo	Imp
Júnior	Nenhuma	Alta	Pequenas tarefas + funcionalidades inteiras com algum auxílio	Seu t
Pleno	2-3 anos	Pouca/baixa	Pequenas tarefas + funcionalidades inteiras + resolução de bugs	Seu t
Sênior	5+ anos	Independente	Funcionalidades inteiras + compartilhar conhecimento + tomadas de decisão	Seu t

(continua...)

Cargo	Experiência	Dependência	Escopo
Staff	8+ anos	Independente	Direcionamento técnico de múltiplos times
Senior staff	Indefinido	Independente	Direcionamento técnico de múltiplos times de especialistas
Principal	Indefinido	Independente	Direcionamento técnico da organização como um todo
Distinguished	Indefinido	Independente	Direcionamento técnico da organização como um todo; contribuições para a comunidade de software; contribuição para a indústria de software
Fellow	Indefinido	Independente	Direcionamento técnico da organização como um todo; contribuições para a comunidade de software; contribuição para a indústria

Cargo	Experiência	Dependência	Escopo
			de software; impacto na sociedade

Referências

- *Salários do cargo de Senior Software Engineer – São Paulo* (GLASSDOOR, c2023): https://bit.ly/glassdoor-salario-senior-sp

- *GitLab Job Grades* (BARNES; WILKINS, c2023): https://bit.ly/gitlab-levels

- *Distinguished software engineer* (THE GITLAB HANDBOOK, 2023a): https://bit.ly/gitlab-distinguished

- *Fellow* (THE GITLAB HANDBOOK, 2023b): https://bit.ly/gitlab-fellow

- *O que é uma startup unicórnio?* (PACETE, 2022): https://bit.ly/forbes-unicornios

Capítulo 4
Os diferentes perfis das pessoas especialistas

A definição exata do que faz uma pessoa especialista em engenharia de software é algo muito difícil. Até por isso, no capítulo anterior, nós não escrevemos em pedra o que exatamente faz cada papel ou qual seu real nível e impacto dentro de uma organização. Mas existem similaridades entre a atuação em diversas empresas e perfis que compartilham suas experiências, por isso eu cataloguei isso aqui como referência.

Além de entender o que faz uma pessoa nesse nível profissional, é interessante também traçar perfis para que tenhamos como visualizar o que realmente fazemos no dia a dia ou como podemos extrair o melhor do modo como nós nos comportamos diariamente. Isso nos ajuda a pensar como utilizar o nosso perfil para conseguir o melhor resultado possível para a organização e também para nossa felicidade profissional.

O nosso perfil seria o nosso modo de trabalhar. Pessoas em um mesmo nível profissional ou cargo realizam mais tarefas relacionadas com o que são boas (ou mais gostam) do que as tarefas em que não vão agregar tanto. Isso se dá por diversos motivos, como **direcionamento da gestão/liderança** ou **busca por visibilidade profissional**, por exemplo.

Quando a liderança nos direciona para algo em que sabe que temos mais conhecimento ou gostamos mais, ela deseja que foquemos em algo onde traremos mais produtividade/velocidade/tração para um time. Mas, quando estamos buscando uma promoção e queremos mostrar que realmente somos bons/boas ou estamos preparados(as) para subir de nível, precisamos focar nos nossos pontos fortes para mostrar valor para a organização.

Claro que não são somente esses fatores que levam alguém a focar em algo, estes são somente dois exemplos de como, por diferentes motivos, nós podemos nos beneficiar de entender qual é o nosso perfil profissional.

Podemos pensar em perfil como *o modo como nos comportamos no dia a dia*. Uma pessoa pode gostar de resolver bug em produção, gostar da adrenalina que isso gera. Outra pessoa pode achar que isso é estresse demais e não gostar dessa ansiedade toda. Outra profissional pode ser extremamente tática dentro do time, trazer as melhores ideias para o futuro do nosso software, mas não ser tão boa ou produtiva codificando quanto a outra que não gosta da parte estratégica e é mais mão na massa.

Perceba: nossa área é criativa, portanto duas pessoas não trabalham iguais, e seu dia a dia pode ser mais bem aproveitado se souber do que realmente gosta ou onde investir seu tempo. Isso evita frustrações e problemas de autoestima, que acabam com a produtividade, como quando uma pessoa que não é tão mão na massa se sente mal por não codificar tão rápido, ou a pessoa mão na massa se sente mal por não conseguir ter uma visão tão de longo prazo quanto a primeira.

Nós podemos evitar essas dores entendendo que a melhor abordagem a seguir é buscar um equilíbrio entre o que fazemos muito bem e o que não nos traz produtividade. No dia a dia, não tem como fugir das tarefas. Em algum momento a pessoa mais estratégica vai precisar codificar e talvez codificar rápido, enquanto o mesmo pode acontecer com a pessoa mão na massa, que será cobrada por uma visão de longo prazo para a saúde do sistema em vez de só sentar e codificar. Precisamos encontrar o caminho do meio.

Duas pessoas fizeram um trabalho maravilhoso em catalogar como nós trabalhamos e como podemos aproveitar melhor as nossas habilidades: Will Larson, em seu livro *Staff Engineer: Leadership beyond the management track*, e Tanya Reilly, no livro *The Staff*

Engineer's Path: A Guide for Individual Contributors Navigating Growth and Change. Nós vamos refletir somente sobre a parte em que falam dos arquétipos de pessoas desenvolvedoras de software, mas ambos são livros excelentes cuja leitura eu também recomendo para entender suas visões, refletir sobre o que eles falam diferente de mim e também para conhecer o mercado internacional e se preparar para o que pode estar por vir no futuro no Brasil.

Eu gostaria de começar pensando no tipo de abordagem de trabalho com a qual nos identificamos. Nós gostamos de focar em problemas específicos de uma área ou tecnologia ou gostamos de arriscar de tudo um pouco e participar de discussões de problemas mais abrangentes de diversos times? Em seguida, vamos conhecer alguns dos arquétipos já conhecidos na nossa área, como tech lead e arquiteto(a), e também refletir sobre esses perfis dentro das nossas empresas. Para cada caso, vamos tentar aprofundar em como é o dia a dia para nos ajudar a refletir.

4.1 Atuação específica ou generalista

Eu adoro as perguntas que Tanya Reilly utiliza em seu livro quando comenta sobre as diferentes abordagens de trabalho:

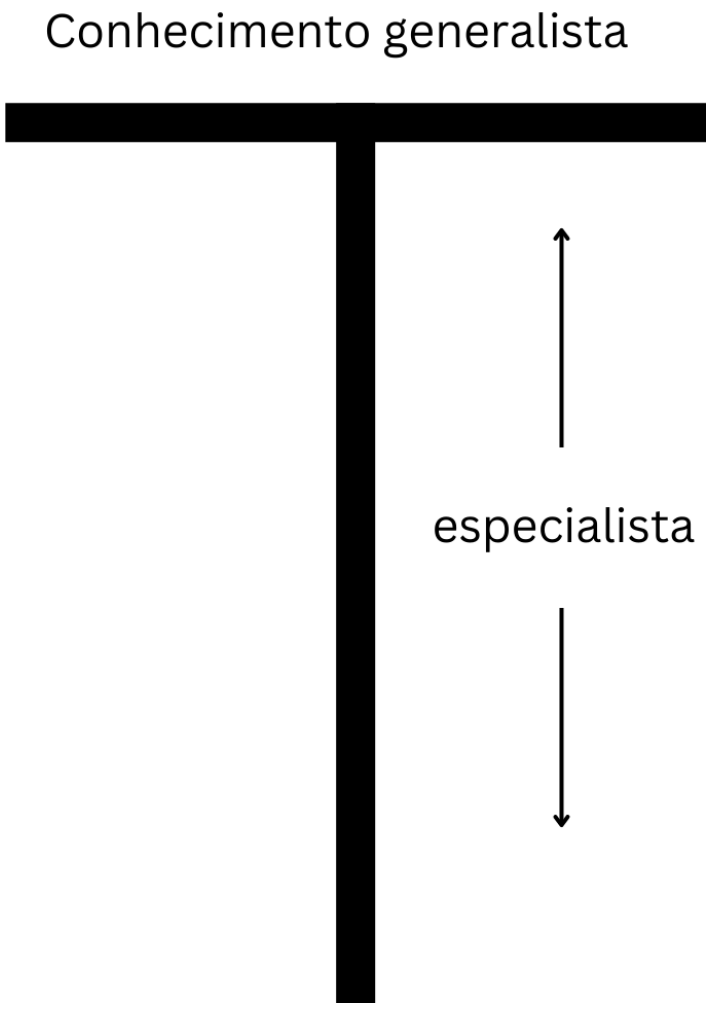
> *"Do you prefer to focus narrowly on a single problem or technology area? Or are you more inclined to go broad across multiple teams or technologies, focusing on a single problem only when it can't be solved without you?"*
>
> Em tradução livre: Você prefere se concentrar em um único problema ou tecnologia? Ou você está mais inclinado a abranger várias equipes ou tecnologias, concentrando-se em um único problema apenas quando ele não pode ser resolvido sem você?

Veja que há uma separação entre alguém que gostaria de ter foco em problemas específicos, como desenvolvimento de aplicativos para celular, por exemplo, e alguém que gostaria de passear entre tecnologias e áreas e focar em uma especialidade somente quando ninguém mais consegue resolver aquele problema. A pessoa especialista em mobile poderia passear entre times, mas sempre resolvendo problemas de mobile. A pessoa que preferiu seguir como generalista também tem uma especialidade, afinal sempre temos algo de que gostamos mais, por isso ela poderia participar de vários tipos de tarefas e discussões além de sua especialização e só focaria nisso quando necessário.

Para entender sobre especialidade versus generalização, eu gosto de citar o caso da pessoa que atua como "dev em T", ou *T-shaped professional*. Esse termo virou moda na nossa área depois de um *paper* da Valve, empresa de jogos muito famosa e que possui um quadro de excelentes pessoas desenvolvedoras, no qual é citada a atuação da pessoa com perfil *T-shaped*. No Brasil, temos o costume de chamar essa pessoa de *full-stack software engineer*. Em outros países, normalmente não temos tanta necessidade de denominar algo, pois os cargos são nivelados como *software engineer* e, internamente, a pessoa atua em sua especialidade ou como generalista.

Em seu *handbook* para novos funcionários, a Valve (2012) define as pessoas com perfil *T-shaped* como:

> *"That is, people who are both **GENERALISTS** (highly skilled at a broad set of valuable things — the top of the T) and also **experts** (among the best in their field within a narrow discipline — the vertical leg of the T)."*
>
> Em tradução livre: *Isto é, pessoas que são tanto **generalistas** (altamente habilidosas em um amplo conjunto de coisas valiosas — o topo do T) como também **especialistas** (entre os melhores em seu campo dentro de uma disciplina restrita — a perna vertical do T).*



Em minha atuação como engenheiro de software, eu preferi focar em me tornar uma pessoa com conhecimento profundo em front-

end web (acessibilidade, segurança no navegador, performance, SEO etc.), mas que também consegue participar de discussões de infraestrutura, arquitetura de software e segurança da informação. Isso não foi uma demanda da gestão ou mesmo intencional de minha parte, foi uma questão de conhecimentos prévios que eu possuía antes de trabalhar como desenvolvedor e que continuei acompanhando no passar dos anos. Eu comecei em tecnologia atuando como técnico em informática, o famoso menino do TI, depois me tornei administrador de redes (infraestrutura e segurança da informação) e, quando entrei na área de programação, gostei muito de discutir de maneira mais abrangente a construção do software (arquitetura). Perceba como temos uma preferência de atuação desde sempre. Só nos falta identificar qual é essa preferência e investir nela.

Não existe um modo fácil de identificar essa preferência ou aquilo no qual realmente somos muito bons/boas. Pode ser que isso aconteça cedo em sua carreira ou somente depois de vários anos, como foi no meu caso. Independente do tempo que isso pode demorar, talvez eu consiga ajudar você a se entender um pouco mais nos próximos tópicos.

A indecisão entre especialização e generalização

O medo de escolher errado e acabar perdendo tempo, de enfrentar situações que nos colocam em desafios que não damos conta ou até mesmo de perder o emprego são exemplos de pensamentos que passam na nossa cabeça na hora de decidir sobre nossas carreiras. Por isso, escolher entre atuar como generalista ou especialista não é uma decisão trivial.

Como eu comentei anteriormente, eu segui pelo caminho da generalização por curiosidade pessoal, não por uma escolha ativa pensando em futuro. Eu sempre procuro me posicionar como desenvolvedor front-end web, porém, no dia a dia, eu também gosto de participar de discussões e tomadas de decisão não relacionadas a isso. Não é incomum eu estar em um projeto de definições de

arquitetura para a construção de microsserviços, assim como em tópicos relacionados com internacionalização da interface, melhorias de performance ou construção e manutenção de *design systems* e também em conversas sobre modelagem de um banco de dados. No meu cotidiano, eu escolhi escrever código no front-end, mas também navegar entre diferentes problemas organizacionais e de projetos de software.

No entanto, eu já tive momentos em que travei e tive dúvidas sobre se estava no caminho certo. Em determinado ponto da minha carreira, eu tive uma conversa com meu diretor sobre o seguinte problema, que estava me afetando bastante: eu não sentia que tinha boas discussões sobre front-end web acontecendo e acreditava que isso era porque eu teria deixado de gostar de trabalhar como front-ender. Na conversa, descobrimos que era muito pelo contrário: eu adoro trabalhar com os tópicos relacionados ao desenvolvimento front-end, inclusive independente de web, mobile ou desktop. Eu adoro a matéria de interação humano-máquina, porém eu não tinha nenhum desafio intelectual para enfrentar naquele momento, o que me deixou estagnado e sem motivação. Parte da emoção de trabalhar com desenvolvimento de software é trabalhar com nossa criatividade; se não temos esse estímulo (por menor que seja), acabamos desanimando com o tempo.

Visto que estava desanimado, pensei "eu gosto de segurança da informação e, se eu trabalhasse com infraestrutura, eu poderia voltar a acessar esses tópicos mais profundamente. Talvez esta seja a hora de trocar de área". Levei esse raciocínio para mais uma conversa e o feedback que recebi foi que, em qualquer área de atuação, chega uma hora em que não temos desafios dentro da empresa onde trabalhamos, e isso não significa que o trabalho como especialista acabou, mas que nós não estamos olhando a imagem como um todo. Faltava-me uma visão geral da empresa, os problemas que ela tinha e como eu poderia trabalhar para resolver isso.

Até aquele momento, eu ainda não me sentia um profissional generalista. Eu acreditava que era um especialista em front-end e somente isso. Além da falta de visão do todo, que meu diretor comentou, eu acreditava que teria que mudar de área (de front-end para outra especialização) por conta de estar desanimado. Na realidade, eu poderia buscar desafio nas outras áreas sem migrar de uma para outra. A partir do momento em que nosso trabalho está realizado e o alinhamento com a liderança é de que podemos dar um tempo daquele assunto, nós podemos buscar outros problemas para resolver.

Assim como Paulo Silveira, CEO e cofundador da Alura, comentou uma vez em uma palestra excelente que assisti em um evento do iMasters (imasters.com.br):

> *Você não é pago para escrever código, é pago para resolver problemas.*

Essa frase me marcou, mas, mesmo tendo ouvido isso antes dessa conversa com meu diretor, meu foco estava em escrever código, e não em descobrir quais eram os problemas que a empresa possuía ou quais seriam os desafios futuros para os quais eu já poderia começar a preparar nossos times para enfrentar.

Perceba que esse raciocínio sobre a visão do todo é independente de atuação como generalista ou não generalista. Naquele momento, eu pude escolher outros problemas para focar, mas eu poderia ter parado para refletir somente sobre o front-end web da empresa. Em todo caso, sempre temos mais desafios do que imaginamos ou que estamos vendo nas tarefas cotidianas.

Independente do caminho que nós escolhemos, no final do dia o nosso trabalho será essa descoberta de quais são os pontos em que a empresa ou os projetos podem melhorar, revisar decisões tomadas para refletir se elas ainda fazem sentido, validar com outros times se

estamos no caminho certo e sempre pensar muito no futuro, não somente no código que escrevemos hoje.

Escolher onde investir seu tempo não será nada fácil, porém é importante levar na mente que o foco deve ser quais problemas você deseja resolver, não com quais tecnologias você deseja trabalhar. Esse tipo de pensamento pode lhe ajudar bastante a conseguir fazer uma escolha.

Vamos refletir um pouco sobre como é a atuação de pessoas especialistas, pensar nos modelos de atuação mais estratégico ou mão na massa. Talvez isso nos ajude a vislumbrar um pouco melhor um possível caminho para o nosso futuro.

Atuação como especialista

A pessoa que normalmente trabalha como especialista em algum ponto específico da engenharia de software é aquela que, apesar das muitas opções de onde investir seu tempo e energia, sempre preferiu focar em uma tecnologia ou área de atuação ao invés de investir em muitas coisas ao mesmo tempo ou em muitas disciplinas ao longo do tempo. Habitualmente, quando trabalhando em um time multidisciplinar, essa pessoa sempre preferiu puxar tarefas relacionadas à sua especialidade, e mesmo quando não tinha nenhuma tarefa cadastrada, trazia novas possibilidades de melhorias, provas de conceito ou discussões relacionadas a esse tema.

> Times multidisciplinares são aqueles nos quais há designers, back-enders, front-enders, mobile, desktop, todo mundo trabalhando junto no mesmo projeto ao mesmo tempo.

Veja, por exemplo, as pessoas especialistas em desenvolvimento de aplicativos para celular (*mobile software engineers*). Quando estão atuando em algum projeto, normalmente a pessoa que escolheu ser especialista em mobile não se envolve na construção direta de uma

API, codificando novos endpoints, modificando o que já existe, integrando esses serviços com terceiros etc. Ela prefere continuar resolvendo problemas relacionados ao desenvolvimento mobile em vez de atacar tarefas não relacionadas a isso.

Ou seja, como especialistas não generalistas, estaremos sempre em discussões relacionadas à nossa área de especialização. Se nosso foco sempre foi front-end ou back-end web, mobile Android, mobile iOS etc., esse será o foco do nosso dia a dia.

Pode acontecer de precisarmos discutir sobre coisas não relacionadas a isso em algum momento, afinal de contas o nosso ambiente de trabalho é imprevisível, porém o foco principal será sempre nossa especialidade.

Exemplos de tarefas de pessoas especialistas:

- Definição da *stack* utilizada para construção das aplicações em sua especialidade;
- Participação em discussões/tomada de decisão sobre a sua especialidade;
- Melhorias em seus ambientes;
- Resolução de problemas específicos de código, que pode ou não ser um bug crítico ou que pode causar um bug crítico, mas que somente você sabe resolver, por ser especialista nesse assunto.

Atuação como generalista

A pessoa com perfil generalista desde sempre acaba se envolvendo com mais do que um assunto dentro do ciclo de vida de construção de um software. Nós sempre temos alguma preferência, mas, mesmo atuando com um tema específico (como back-end ou mobile), acabamos estudamos outras coisas não relacionadas. A pessoa generalista, mesmo que não trabalhe com essas outras coisas no seu dia a dia, acaba estudando, fazendo provas de conceito e outras experiências por gostar se aventurar em tarefas fora do seu escopo. Essa pessoa costuma puxar tarefas não

relacionadas com sua atribuição principal desde sempre ou participar de discussões relacionadas a outras frentes, mesmo que não vá colocar a mão na massa, e consegue gerar valor com isso.

Como comentei anteriormente, mesmo sendo profissionais generalistas (*full-stack*), teremos a nossa especialidade, pois todo mundo tem algum assunto que domina mais, porém ele não será o único foco do nosso dia a dia. Em algum momento, podemos estar em uma reunião fora da nossa especialidade, mas que tem alguma relação direta ou indireta por cuidarmos de um software ou de um ecossistema completo que envolve várias tecnologias ou assuntos. Normalmente não seremos colocados em qualquer tarefa aleatória, pois, apesar de generalistas, temos conhecimento limitado, não somos máquinas. Resolveremos nossos problemas exclusivos e auxiliaremos ou desenvolveremos para outras especialidades nas quais temos interesse.

Na minha experiência profissional, a maioria dos profissionais *full-stack* que conheci eram pessoas de back-end com um pouco de conhecimento em front-end, mas também conheci algumas pessoas que são excelentes em front-end e se arriscam no back-end. Não existe uma regra para dizer quem consegue ou não ser *full-stack*, é basicamente uma questão de interesse pessoal e não tem nenhuma relação com a pessoa ser melhor do que alguém não generalista por conseguir atuar em várias pontas do projeto.

Exemplos de tarefas de pessoas generalistas:

- Todas as tarefas de uma pessoa com especialização em um tema;
- Participação em discussões/tomadas de decisão fora de sua especialidade, mas que tenham relação com seu conhecimento. Ex.: uma pessoa back-end que saiba de infraestrutura pode ajudar outros times com front-end que precisam de apoio em suas definições de infra;
- Apoio para debug de bugs não relacionados a sua especialização, mas que tenham algum ponto de conexão — por

exemplo, a pessoa back-end que tem bom conhecimento em Kibana e que auxilia o time de mobile a entender um bug.

4.2 Atuação estratégica ou mão na massa

Outro ponto que será complexo de refletir em nossa carreira será sobre trabalhar em uma posição de um modo mais estratégico ou mais mão na massa. Ou seja, ser a pessoa que participa de muitas reuniões pensando no futuro, escreve documentações de arquitetura e puxa discussões sobre assuntos relacionados, ou aquela que está mais focada em escrever o código e realizar as tarefas que identificamos juntos nessas reuniões.

Os dois perfis possuem experiências parecidas, mas o modo de trabalho muda bastante. Nós podemos, por exemplo, ter uma pessoa mão na massa que gosta de pegar as histórias/tarefas prontas no Jira ou outro software de gestão de projetos e escrever código no modo turbo focando nas melhores práticas de mercado e outros detalhes de implementação que só ela consegue se lembrar. Do outro lado, temos a pessoa que não gosta tanto assim de escrever código, mas adora definir processos, convenções, boas práticas e afins e documentar isso para que outras pessoas desenvolvedoras sigam o seu racional.

Muita gente acredita que um profissional a nível de especialista não deveria ser tão mão na massa, porém existe, sim, espaço para os dois perfis. Em grandes empresas, a pessoa que gosta de escrever código está mais focada em criar ferramentas que facilitem o dia a dia de outras pessoas desenvolvedoras. Ela não está entregando novas funcionalidades em um produto, como qualquer outra pessoa desenvolvedora conseguiria, ela está escrevendo o código de um módulo que a empresa inteira vai utilizar para agilizar o trabalho no desenvolvimento de produtos, criando ou melhorando ferramentas de geração de código automatizadas, trabalhando próxima aos times

para descobrir como poderia criar algo que facilite suas vidas, resolvendo bugs em produção que outras pessoas não conseguiriam resolver no mesmo tempo hábil ou até mesmo codificando o próximo banco de dados que a empresa vai utilizar.

Outras pessoas acreditam que quem trabalha em posições mais estratégicas não tem tanto impacto quanto quem está escrevendo o código no dia a dia, pois fica confabulando sobre um futuro que pode ou não acontecer. Mas essas pessoas constroem o alicerce forte para garantir a sustentação da empresa por muito tempo. É muito complexo discutir a arquitetura de um software que precisa permanecer no ar durante anos. Não é uma conversa que acaba em 45 minutos ou com um documento de arquitetura, assim como a definição das práticas que os times vão seguir, tecnologias que serão utilizadas, como essas tecnologias serão utilizadas dentro da empresa e como documentar tudo isso, criar templates para garantir as construções certas, observar pipelines ou a saúde dos softwares e sempre pensar no futuro. O exercício de pensar no futuro de um negócio não é trivial ou cotidiano como escrever o código para uma nova funcionalidade do produto, que é um pedaço de um ecossistema dentro da empresa.

Enfim, independente do que as outras pessoas pensam, nós precisamos tomar uma decisão sobre nosso futuro profissional. Vamos atuar mais mão na massa ou de forma estratégica? O que a empresa onde estamos agora consegue nos proporcionar a respeito dessa experiência? Precisamos analisar os cenários para que as experiências individuais das outras pessoas não atrapalhem nossa tomada de decisão.

Atuação mão na massa

A pessoa mais mão na massa é aquela que, antes mesmo de se tornar sênior era considerada a tal da *ticket burner* (queimador de tarefas do Jira ou outro software de gestão de projetos). Aquela pessoa que entrega a maior quantidade de tarefas em uma sprint, por exemplo. Não estamos falando de qualidade ou dizendo que

essa pessoa é mais produtiva. Muitas pessoas que têm esse perfil costumam entregar muito código ruim durante os primeiros anos de carreira, mas com uma boa mentoria começam a melhorar a qualidade e segurança de suas entregas. Mas a produtividade não necessariamente está em escrever código: às vezes se trata de saber quando não escrever, e essa pessoa também aprende isso e continua sendo a que gosta de focar em escrever código.

Normalmente essa pessoa acaba caindo em times de plataforma ou em experiência da pessoa desenvolvedora (*developer experience*), pois consegue pensar e escrever códigos que serão reutilizados por outras pessoas da maneira mais eficiente para que elas não precisem escrever código.

Exemplos de tarefas de pessoas mão na massa:

- Ajudar um time que está iniciando um projeto a codificar rapidamente as primeiras entregas;
- Desenvolver e manter módulos críticos que sustentam os outros times dentro da empresa;
- Desenvolver *scaffoldings* e outros modos de geração de código, como ferramentas de linhas de comando (CLI - Command Line Interface), que ajudam outras pessoas desenvolvedoras no dia a dia;
- Assumir bugs críticos que precisam de solução de código rápida.

Atuação estratégica

A pessoa com atuação mais estratégica é aquela que sempre gostou de discutir o futuro das aplicações em um nível mais alto (considerando baixo nível o contato direto com código e alto nível o contato maior com arquitetura). É comum que essas pessoas também tenham perfil de compartilhamento de conhecimento e considerem extremamente importante a produção e manutenção de documentações, normalmente tomando a frente disso.

Normalmente essa pessoa participa de mais reuniões envolvendo vários times e várias frentes da empresa com o objetivo de extrair requisitos que se encaixem na construção ou sustentação da arquitetura dos sistemas da empresa.

Exemplos de tarefas de pessoas com atuação estratégica:

- Ajuda um time que está iniciando um projeto a planejar a arquitetura do sistema, a integração com outros softwares e times dentro da empresa — e documentar isso;
- Está sempre em reuniões com pessoas de produto, lideranças técnicas dos times e outras pessoas seniores para ajudar a planejar a construção do software trazendo sua visão do todo;
- Normalmente define as estratégias de desenvolvimento de software dentro das empresas, por isso constrói automações e templates que possam auxiliar os outros times a seguirem essas convenções;
- Atua apoiando as lideranças a monitorar aplicações e trazer ideias para melhorias de qualidade.

4.3 Arquétipos em carreira de especialistas

Algo interessante apresentado por Will Larson no seu livro *Staff Engineer: Leadership beyond the management track* e também em seu repositório público (staffeng.com) são os **arquétipos de pessoa desenvolvedora**. Não estamos falando de assumir o arquétipo de Cleópatra, mas ainda sobre os perfis de pessoas desenvolvedoras em nível de especialistas.

Além do que já comentei sobre os perfis, Will apresenta uma visão sobre as atribuições mais comuns de pessoas especialistas — apesar de que muitas podem estar em um arquétipo e de vez em quando agir como outro.

Ele comenta sobre os seguintes perfis (em tradução livre):

- **Tech lead:** orienta a abordagem e execução de uma determinada **equipe**. Eles fazem parceria estreita com um único gerente, mas às vezes fazem parceria com dois ou três gerentes dentro de uma área específica. Algumas empresas também têm uma função de gerente de liderança técnica, que é semelhante ao arquétipo de líder técnico, mas existe na trilha de gerente de engenharia e inclui responsabilidades de gerenciamento de pessoas;
- **Arquiteto(a) de software:** é responsável pela direção, qualidade e abordagem dentro de uma área crítica. Combina conhecimento profundo de restrições técnicas, necessidades do usuário e liderança no nível da organização;
- **Resolvedor(a):** se aprofunda em problemas arbitrariamente complexos e encontra um caminho apropriado para seguir em frente. Alguns se concentram em uma determinada área por longos períodos, outros pulam de ponto em ponto conforme guiados pela liderança organizacional;
- **Mão direita:** estende a atenção de nível executivo, emprestando seu escopo e autoridade para operar organizações particularmente complexas. Fornecem largura de banda de liderança adicional para líderes de organizações de grande porte.

Tech lead é o perfil mais padrão dentro da carreira de especialistas, pois é um cargo que já é antigo dentro da área de engenharia. Arquiteto(a) não é novo, mas razoavelmente mais difícil de encontrar formalmente como um cargo dentro das organizações, mas também encontramos muitas pessoas seniores e tech leads com este perfil. A pessoa resolvedora se assemelha muito com o que comentei sobre pessoas especialistas mais mão na massa, e a pessoa mão direita, com a pessoa mais estratégica.

Conclusão

É muita informação para absorver e também muita coisa para pensar antes de tomar uma decisão, mas saiba que nós não teremos as atuações escritas em pedra, será sempre aquela busca por equilíbrio em nossas funções.

Uma pessoa especialista em um tema pode ser convidada para participar de tomadas de decisão ou construções não relacionadas a sua especialidade; pessoas mão na massa podem acabar sendo chamadas para a discussão sobre padrões e definições de futuro dos sistemas da empresa. Essas pessoas são especialistas e, apesar de suas preferências, atuam como especialistas em engenharia de software e isso é abrangente e mutável durante o dia a dia.

Em nossa trilha profissional, é importante usar as palavras ditas até aqui para nos ajudar a entender nosso perfil e extrair o melhor dele. Assim que entendemos quem realmente somos, conseguimos buscar boas referências para acompanhar, assim como conversar com nossa gestão ou liderança para direcionar nossa atuação e encontrar o ponto de equilíbrio.

Espero que este capítulo tenha ajudado você a se entender um pouco mais. Nos próximos capítulos, continuamos aprofundando sobre o tema.

Referências

- *Handbook for New Employees* (VALVE, 2012): https://bit.ly/valve-handbook-cdn

- *T-shaped Skills* (WIKIPEDIA, c2022): https://en.wikipedia.org/wiki/T-shaped_skills

- *Staff archetypes* (LARSON, c2023): https://staffeng.com/guides/staff-archetypes/

- *Staff Engineer: Leadership beyond the management track* (LARSON, 2021): https://staffeng.com/book/

- *The Staff Engineer's Path: A Guide for Individual Contributors Navigating Growth and Change* (REILLY, 2022): https://amzn.to/3NZyCBF

Parte 2: Prática

Capítulo 5
Entendendo a parte prática da especialização

Agora que nós já sabemos o que realmente faz uma pessoa especialista em engenharia de software, entendemos todas as nuances e possibilidades dentro dessa trilha profissional e também aceitamos que a área não segue um padrão muito bem definido — e que isso pode tornar nosso trabalho desafiador —, podemos finalmente seguir para a parte prática.

Quando eu decidi seguir para a área de especialista, foi difícil encontrar boas referências do que eu deveria fazer para estar pronto para assumir essa posição. Mesmo com a ajuda de bons gestores durante minha carreira até este momento, ainda faltavam dicas que poderiam tornar até mesmo as reuniões de feedback com a gestão mais fáceis.

Nesta parte do livro, vamos seguir o caminho até a consolidação como especialista. Vamos passar pelos vários estágios dessa transição, refletindo sobre se realmente estamos no momento de carreira em que podemos fazer esse movimento (não existe o momento ideal, mas existem assuntos que precisamos levar em conta antes de seguir), pelo momento em que alinhamos como será o processo junto com nossa gestão, além de vermos algumas dicas para quando assumirmos esta posição.

Aqui eu reúno sugestões que recebi de colegas de trabalho, feedback das minhas várias lideranças, assim como dicas que eu aprendi em outros momentos da carreira, fora da área de tecnologia, mas ainda assim muito úteis para qualquer profissional.

Espero que as reflexões trazidas aqui ajudem a sua transição a acontecer de uma maneira mais suave e tranquila, bem como

preparem você para continuar seguindo para as próximas etapas e enfrentar os próximos desafios com mais segurança, pois, quanto mais crescemos, maiores são os problemas que precisamos resolver. Se não estivermos muito bem munidos de informação e preparo, podemos cair direto no caminho para o *burnout*. Atuar como especialista foi o que me motivou ainda mais a continuar nessa área, por isso torço que seja tão motivador para você quanto foi para mim.

Não se preocupe se você não sabe priorizar sua agenda, se nunca teve uma conversa sobre carreira com sua liderança ou mesmo se não teve a oportunidade de planejar sua evolução profissional. Nos próximos capítulos, vamos explorar não somente a possibilidade da promoção, mas também algumas boas práticas para conseguir dar conta do trabalho novo.

Capítulo 6
Colocando os pés no chão

Planejar é parte importante de nossas vidas. Quando nós criamos uma estratégia e pensamos em objetivos de curto e longo prazo, a nossa chance de falhar diminui, assim como as possibilidades de mudanças pelo meio do caminho são facilitadas. Se ficarmos somente no universo das ideias e nunca colocarmos em prática o trabalho que precisamos realizar para alcançar isso, vamos patinar na lama até desistir dos nossos sonhos. Porém, ao cravar em pedra uma ideia única, sem flexibilidade de mudanças durante o percurso, podemos estar em um caminho direto para a frustração profissional.

As coisas mudam bastante em nossa história profissional. Em determinado momento podemos querer fazer de tudo; em outro instante, desejamos focar em uma única coisa. Nossos planos precisam ser flexíveis para atender a essa flutuação que, com certeza, vai acontecer ou está acontecendo neste momento.

Antes de seguirmos de fato para o planejamento de nossas carreiras depois de sênior, eu quero ter uma conversa séria e honesta sobre algumas questões que foram impactantes não só na minha carreira, mas também nas das pessoas que eu mentoro e colegas de profissão, tais como a ansiedade pelo próximo cargo e os riscos de assumir papeis quando ainda não estamos realmente prontos.

6.1 A ansiedade pelo próximo cargo

Mesmo que planejemos uma carreira onde temos pequenos passos para seguir e que já tenhamos o entendimento de que o crescimento vem com o tempo, a ansiedade é algo que nos consome. Quando em fase de estágio ou como auxiliares, queremos nos tornar juniores

logo; quando juniores, queremos a promoção para pleno; quando plenos, queremos correr para o cargo de sênior. Isso acontece não somente pela vontade de aumentar nosso salário — afinal de contas, dá para aumentar o salário sem precisar subir para o próximo nível, mudando totalmente nosso escopo e até mesmo como recebemos cobrança da liderança direta e da gestão. Às vezes é só pelo ego, pelo fato de mudar o nosso status social e também para receber alguns biscoitos no LinkedIn.

É algo totalmente compreensível sentir-se valioso por conta de ter um cargo maior, pois é o que aprendemos socialmente. Nós enxergamos estes sinais de status e valor social o tempo todo em nosso dia a dia e em nossa formação como seres humanos. A pessoa em papel de executiva(o)-chefe de uma empresa é quem é mais respeitada dentro da organização, e a pessoa do próximo nível hierárquico é quem devemos obedecer sem questionar. Fugindo da estrutura organizacional, no nosso dia a dia temos como modo de pensar que, quem tem a melhor casa, o carro mais caro, o celular da moda e outros bens de consumo, é a pessoa que se deu bem na vida.

É por conta desse contexto que estamos sempre querendo buscar o próximo nível, seja por dinheiro, seja para conquistar respeito. O ser humano está sempre buscando validação do grupo. Porém, isso pode ser um problema quando refletimos que não somos naturalmente criados para o estilo de vida em que vivemos. Esse tipo de comportamento é o que nos deixa extremamente ansiosos(as), levando a transtornos como a ansiedade generalizada, depressão, *burnout* e outros males que nos assolam neste século.

> *Disclaimer*: eu chamo de *ansiedade pelo próximo cargo*, porque foi como minha terapeuta e eu conseguimos classificar o que eu sentia. Porém esse não é um termo clínico e, se você quiser replicar isso de maneira mais científica, pode só chamar de ansiedade mesmo.

Essa ansiedade toda pode (e vai) nos colocar em diversos apuros, como assumir uma posição para a qual ainda não temos preparo ou trocar de emprego por uma empresa péssima acreditando que nunca receberia reconhecimento no emprego anterior, por exemplo. Por isso, precisamos continuar nossa reflexão.

6.2 O tempo certo para receber uma promoção

Muitas das pessoas a quem eu presto mentoria, assim como a própria comunidade de programação, levantam questões como: Qual é a hora certa para receber uma promoção? Existe um tempo mínimo de trabalho em um cargo antes de ir para o próximo? O que define que já estamos prontos para o próximo nível?

Não existe um tempo certo para receber uma promoção. Existem pessoas em nível pleno que estão atuando como seniores e também existem pessoas juniores que estão assumindo responsabilidades que deveriam ser do próximo nível. Se eu estou em um time onde a pessoa nível pleno está atuando como sênior, logo eu gostaria que essa pessoa fosse promovida.

Nós vamos acumulando conhecimento que nos faz atuar de maneira diferente de acordo com quanto mais contato tivermos com projetos diferentes e situações complexas. Não se trata somente de escrever código, mas como você escreve, a qualidade do seu trabalho. Mas também não é somente sobre código: a sua postura profissional conta mais para uma promoção para sênior do que quantas *features* você entrega.

O mais importante de se levar em consideração é: **Quão consolidados estamos no cargo em que atuamos neste momento?** Se você é uma pessoa júnior, você consegue garantir que já fez tudo o que alguém nessa posição deveria fazer? Se você é pleno ou sênior, você acha que já não tem mais nada de novo que

poderia ser apresentado para você dentro das atribuições que lhe foram dadas?

Para exemplificar melhor o que estou tentando dizer, vou recorrer novamente às tabelas. Vamos imaginar que estamos trabalhando em uma empresa cujas atribuições estão muito bem definidas para cada cargo. Essas atribuições estão na tabela a seguir:

Cargo	Escopo	Supervisão	Conhecimento técnico
Júnior	Time	Direta	Básico
Pleno	Time	Somente em tarefas complexas	Intermediário
Sênior	Time	Não precisa	Avançado

Para fins de exemplo, vamos considerar que, nessa empresa, o conhecimento técnico seria dividido desta forma:

- **Básico:** conhece a sintaxe básica da linguagem de programação, conhece comandos básicos do Git e uso de Terminal; sem supervisão, pode colocar bugs em produção;
- **Intermediário:** tem conhecimento profundo da linguagem de programação, utiliza comandos avançados do Git e do Terminal, consegue entregar tarefas simples em sua *stack* sem pedir ajuda;
- **Avançado:** tem conhecimento ainda mais profundo da linguagem de programação (como utilizar a linguagem da maneira mais eficiente possível, evitar os problemas de implementação da linguagem e mais), consegue resolver problemas de versionamento complexos sem dificuldade,

domina o uso de ferramentas de produtividade, entrega tarefas complexas sem nenhuma supervisão.

Pensando somente nesses tópicos, conseguiríamos definir um caminho a seguir para subir de níveis dentro desta organização. A pessoa júnior que já consegue finalizar tarefas sem supervisão, mas ainda coloca bugs em produção, não estaria pronta para subir de nível. A pessoa pleno que consegue entregar tarefas mais complexas, mas ainda não consegue ter certa produtividade no versionamento de código, ainda não estaria pronta para uma promoção. A pessoa sênior que consegue entregar tarefas complexas, mas ainda não domina seu ambiente de desenvolvimento, ainda não estaria pronta para uma promoção.

Claro que isso é ilustrativo. No dia a dia de trabalho, as coisas não funcionam tão rigidamente. As pessoas podem demorar mais ou subir de nível para o próximo cargo sem atingir todos os requisitos de um checklist. O objetivo de termos uma tabela, slide ou qualquer documentação da empresa para entendermos as nossas atribuições é simplesmente para evitar a falta de entendimento do que é esperado do nosso trabalho.

Visto que temos objetivos de curto e longo prazo para atingir, conseguimos controlar a nossa ansiedade pelo próximo cargo se buscarmos feedback constante sobre nossa atuação e também manter o alinhamento direto com a nossa liderança e gestão. Caso a empresa não tenha um documento explicando cada posição, podemos alinhar com a nossa gestão e buscar atingir esses objetivos como modo de entendermos se realmente estamos prontos para subir de nível. Isso pode acontecer rápido ou devagar, pois depende do nosso contexto de trabalho, momento de vida e de outras variáveis sobre as quais nós não temos controle.

O importante é manter a busca por feedback dos nossos pares para validar o que estamos fazendo e o que achamos que estamos fazendo, com a visão do time e superiores. Podemos fazer isso durante as conversas com nossa gestão, mas também podemos

tomar a atitude proativa de agendar bate-papos com as pessoas do nosso time para recebermos este retorno. Isso será valioso tanto para sabermos quando estamos prontos quanto para direcionar onde devemos investir nossa energia.

6.3 Promoções e tempo de trabalho dentro de uma empresa

Muita gente já deve ter ouvido em uma entrevista de emprego ou durante o onboarding em uma nova organização sobre como as promoções acontecem com base em meritocracia, sobre como as pessoas que "vestem a camisa" da empresa recebem promoções infinitamente e nunca param de receber reconhecimento financeiro e destaque dentro da empresa por trabalharem mais do que as outras que estão na média.

Esse tipo de discurso acaba nos iludindo, achando que vamos receber um aumento cada vez que a empresa apresentar bons resultados financeiros, ou que todo ano vamos receber uma promoção caso nos esforcemos mais do que a média. Esse discurso é um dos ingredientes da fórmula para o *burnout*, mas, fora este problemão, está longe de ser verdade.

Não existe uma regra sobre de quanto em quanto tempo nós recebemos um aumento ou uma promoção, a não ser o que é obrigatório por alguma regulamentação ou leis nacionais ou locais. Existem organizações onde as pessoas estão sempre crescendo, e outros lugares onde passam anos no mesmo cargo ou em um cargo mais baixo fazendo exatamente o mesmo que o cargo mais alto. É algo extremamente complexo.

Vamos analisar esse assunto com uma visão mais crítica e materialista das coisas. Pode soar até meio desanimador, mas pensar nessa realidade é importante para que não criemos

expectativas demais com o marketing do recrutamento e seleção e acabemos acumulando decepções e tristeza nesse mercado ao invés de conquistas e boas realizações.

Promoções fazem parte da estratégia da empresa para conseguir manter seu quadro de funcionários. Afinal de contas, se você está há muito tempo recebendo somente um pequeno aumento todo ano, provavelmente vai querer trocar de empresa, pois os preços dos bens de consumo e itens importantes para nossa existência sobem fora da empresa — se nosso salário não acompanha, o desespero toma conta. E, claro, todo mundo quer receber reconhecimento pelo bom trabalho ou simplesmente ganhar mais dinheiro para conquistar seus sonhos pessoais.

Além de serem ferramentas estratégicas de motivação e extremamente importantes para manter a qualidade de vida dos funcionários, aumento de salário e promoções aumentam o custo operacional de uma organização. Se a empresa não tem lucro suficiente para pagar essas contas, o negócio fica insustentável para o longo prazo.

Empresas são um aglomerado de complexidade que temos que lidar diariamente em diversos níveis. Se você está em um cargo operacional, como engenharia de software, está lidando com a produção que vai gerar lucro para a empresa. Porém, para essa produção acontecer, a empresa possui contas a pagar com fornecedores, impostos, salário de outros times, infraestrutura para que você faça o seu trabalho e muito mais. Para complicar ainda mais, dependendo do estágio e tamanho da sua organização, ela pode ter capital aberto, com dezenas, centenas e talvez milhões de investidores, que vão cobrar que a organização tenha lucro para receber sua fatia do bolo que foi gerado com o seu trabalho.

Meritocracia não existe, porque as coisas não dependem só do seu esforço. O mundo é muito mais complexo do que isso. Enquanto você faz o seu trabalho buscando uma promoção, existem várias camadas de problemas que a sua gestão terá de enfrentar para lhe

entregar esse reconhecimento. Você pode ter atingido todos os requisitos e mesmo assim não poder subir de cargo porque:

- Não tem vaga aberta para um nível acima do seu;
- Outras pessoas em outros times trabalham em projetos que geram mais valor para a empresa e as vagas serão destinadas para elas;
- A empresa não tem tamanho suficiente para ter mais especialistas (ou seniores, ou plenos, ou até juniores, isso se encaixa para qualquer nível);
- A empresa precisa conter custos operacionais para garantir um bom resultado financeiro e atrair mais investimento;
- A gestão não tem capacidade de negociação para defender você na hora de negociar cargos com a diretoria;
- Você não tem "tempo de casa", e uma promoção faria outras pessoas desanimarem;
- Você não tem tempo de mercado, e a empresa possui isso como regra de promoção.

Esses são apenas alguns dos muitos motivos possíveis. Eu tive contato com esses tópicos enquanto participava das reuniões de calibração (quando estamos analisando a performance dos times) junto com a gestão de algumas empresas onde trabalhei.

Para que isso não se torne um problema, precisamos manter a mente aberta e sempre refletir sobre a realidade ao nosso redor. Podemos verificar com a nossa liderança qual é o tempo médio em que as pessoas são promovidas, se a empresa está crescendo, se o negócio está dando lucros (empresa lucrar não é o mesmo que empresa crescer) e até mesmo se a gente realmente tem o que fazer se receber uma promoção.

Caso a liderança não tenha essas respostas ou não queira responder, podemos analisar com mais atenção o posicionamento da empresa para o mercado financeiro. Podemos aprender a analisar os dados de relações com investidores, uma carta aberta que as empresas fazem mostrando os resultados financeiros, planos para o futuro e

muito mais. Se a empresa não tem capital aberto, analisamos o posicionamento da marca, se ela está conseguindo penetrar no mercado e crescer em clientes ativos, parceiros comerciais e mais. Com essa realidade em vista, podemos começar a aceitar melhor quando não recebemos uma promoção por tempo de casa, em um ciclo de avaliações ou mesmo quando vamos atrás de pedir esse reconhecimento de nossa gestão. Nem sempre a gestão pode abrir todas essas informações, nem sempre a gestão tem essas informações. O importante é ter isso em mente durante a nossa vida profissional.

Se a empresa nos paga bem, se estamos recebendo desafio técnico e intelectual, se temos bons benefícios e qualidade de vida com equilíbrio entre trabalho e vida pessoal, entender esses pontos nos ajuda a conseguir esperar o nosso momento e não querer pular de empresa em empresa atrás de uma promoção, o que pode ser um grande problema se trocarmos algo bom por algo ruim. Mas conhecer isso também nos ajuda a entender quando estamos sendo enrolados(as) pela empresa e saber que é a hora certa de sair dali para um lugar melhor.

6.4 O perigo de assumir uma posição para a qual não estamos preparados

Em muitos momentos, e por diversos motivos, podemos acabar onde não deveríamos estar (ainda). Talvez uma promoção por tempo de casa, talvez aquela indicação forte que conseguimos antes de entrar no emprego ou só porque bateu a ansiedade e pedimos promoção ou pulamos para outro emprego. Em qualquer caso desses, assumir uma responsabilidade antes de estar pronto pode nos causar muitos problemas.

Em um momento da minha vida eu precisei direcionar meus esforços para aumentar meu salário o máximo que eu conseguisse em

qualquer nível ou qualquer empresa. Isso não é errado ou certo, cada pessoa sabe onde sente mais dores. Porém esse comportamento me trouxe a ansiedade não pelo próximo aumento, mas pelo próximo cargo, afinal de contas, uma promoção nos traz um retorno financeiro maior do que um aumento salarial padrão. Desde que assumi essa postura, foi questão de tempo até eu ter um *burnout*. Para ser mais específico, dois anos e meio.

Eu deixei uma empresa muito legal, onde o time me apoiava e eu tinha um crescimento planejado com todo o apoio da minha liderança, pelo aumento salarial e a promessa de trabalhar com uma *stack* que poderia me dar ainda mais retorno financeiro no futuro. A parte do aumento e o retorno financeiro futuro deu muito certo, contudo eu tive o ponto negativo de minha saúde mental ir para o ralo por conta das duas empresas onde trabalhei depois dessa troca.

Nós sempre temos riscos e benefícios em fazer alguma troca. Seja trocar de tecnologia, seja mudar de cidade ou qualquer outra coisa que pode mexer em nossas vidas. Precisamos medir cada uma com muita atenção e garantir que tenhamos um futuro pela frente para conseguir aproveitar essas trocas. Uma mudança sem planejamento e visão crítica de cada detalhe pode acabar com nosso futuro.

O que aconteceu comigo foi o fato de que eu ainda não tinha tanta autonomia ao codificar quanto uma pessoa nível pleno precisa. Por isso eu precisei acelerar muito meu ritmo de estudos e trabalho para dar conta das demandas que apareciam. Eu também não tinha maturidade para receber feedbacks críticos, e eles aumentam bastante conforme subimos nosso nível profissional. Foi a fórmula precisa para explodir.

Nesses casos, um dos problemas que podemos enfrentar é a nossa própria saúde mental ir embora pouco a pouco. A cobrança de cada nível aumenta exponencialmente. Quanto maior seu cargo, maior deve ser seu impacto, a qualidade do seu trabalho, o retorno financeiro que você traz para a empresa. Se não estivermos

realmente prontos para o próximo cargo, isso coloca um peso enorme em cima de nós.

Mas isso não acontece somente por nossas atitudes e ansiedades. Eu também já vi pessoas que foram convidadas a assumir posições para as quais ainda não estavam preparadas sem a possibilidade de dar um passo para trás por sua liderança. Afinal de contas, a promoção de um membro de sua equipe é métrica de sucesso para um bom líder, mas estamos em um mundo capitalista, onde nossa líder também sofre pressão pela produtividade e crescimento do time, o que faz com que essa pessoa também possa cometer o erro de criar um crescimento artificial de seu time.

Outro caso que pode acontecer é sua empresa ser comprada e você se tornar parte do quadro profissional de outra empresa assumindo um cargo maior do que o que você tinha por conta de algum acordo do quadro executivo. Isso é bem comum no mundo das startups, onde, muitas vezes, os investidores forçam essas compras para garantir a eficiência do seu portfólio de investimentos. Uma startup pode ser absorvida por outra por não estar gerando lucro, mas as pessoas querem continuar com os times em que estão. Neste momento, podemos cair em situações difíceis para a nossa carreira, mas também pode ser a oportunidade que nos faltava para recebermos a merecida promoção e a empresa não tinha caixa suficiente para isso.

Além disso, a falta de preparo pode trazer sérios riscos para a empresa, e decisões ruins começam a afetar a nossa autoestima pouco a pouco. Seja o aumento do custo dos sistemas ou pela escolha de uma tecnologia que pode ser o ponto crítico da empresa, que vai fazer ficar mais difícil de incluir pessoas novas no time ou de dar manutenção em um software legado, existem muitas variáveis que nós aprendemos com a experiência e a vivência com outros profissionais mais qualificados e experientes que nós. Neste caso, se você tiver uma excelente autoestima e não perceber que suas decisões são péssimas, o risco que está correndo é o de perder o emprego e, se realmente não estiver preparado(a), não conseguir

passar em vagas de emprego para o mesmo cargo, tendo que rebaixar seu salário ou demorando demais para se recolocar no mercado de trabalho.

Mas também existem casos onde a liderança acredita que estamos prontos para o próximo nível, mesmo não atingindo todas as expectativas, pois ele(a) sabe que temos potencial que ainda não foi explorado. Nesses casos, normalmente temos o apoio direto dessa liderança caso assumamos a próxima posição, para ter uma transição suave e com feedbacks produtivos para conseguir estabilidade rapidamente.

Em todos os casos, seja uma promoção forçada por nós, pela nossa liderança, por uma compra ou qualquer outra coisa que pode acontecer (e muitas podem acontecer), nós precisamos sempre avaliar se estamos realmente prontos para assumir certas responsabilidades, pois quanto maior o nosso cargo, maior será a cobrança sobre o nosso trabalho, e talvez nós não tenhamos tempo para absorver essa mudança e corremos o risco de cair em grandes problemas.

6.5 Ocasionalmente estamos em cargos abaixo do nosso nível, mas não sempre

O ego é nosso principal inimigo no ambiente de trabalho. Seja para nos colocar para cima ou para baixo, o nosso subconsciente tem o poder de nos atrapalhar bastante. Em alguns momentos estamos com a autoestima tão baixa que, mesmo com toda capacitação do mundo, experiência profissional e todos os astros se alinhando para que nosso dia a dia seja bom, ainda caímos na síndrome do impostor, que é quando não acreditamos estar no lugar certo. Porém nem sempre é assim.

Em determinados momentos, nós acreditamos bater todas as metas, requisitos e ter a maior capacitação do mundo para assumir um cargo, mas, na realidade, estamos abaixo (ou muito abaixo) disso. Enquanto de um lado temos a síndrome do impostor, do outro temos o efeito Dunning–Kruger.

> O efeito Dunning-Kruger é o viés pelo qual pessoas com baixa habilidade em uma tarefa superestimam sua habilidade. É o completo oposto da síndrome do impostor, onde a pessoa extremamente qualificada não se sente apta de fato.

Manter o equilíbrio em nossas vidas é um eterno exercício de avaliação de tudo ao nosso redor. Para a carreira não seria diferente.

Você pode realmente estar em uma empresa que não valoriza o seu trabalho e não está reconhecendo o seu potencial, mas essa hipótese não pode ficar somente em sua cabeça. Você precisa buscar informação para validar se está certo(a) ou se realmente precisa mudar algo antes de conquistar o próximo nível.

Não só de nível técnico vive a pessoa desenvolvedora de software. Você pode ter uma excelência técnica, porém sem atitudes que demonstrem que você teria produtividade no próximo cargo. E o inverso também acontece, quando você já tem a postura que precisa para subir de cargo, mas ainda não atingiu maturidade técnica suficiente.

Para evitar qualquer problema, neste momento seria ideal procurar alguém em um cargo maior que o seu e com mais experiência que você, da mesma empresa ou não, além de buscar a validação da sua liderança e pares. Muitas vezes nossos pares comentam que nós estamos fazendo mais do que o nosso cargo requer, mas, quando eles não falam nada, é um sinal de atenção. Alguém de fora pode nos ajudar a colocar os pés no chão ou mesmo entender de uma vez por todas como estamos profissionalmente.

6.6 Quando a promoção não passa de promessa

Até aqui refletimos bastante sobre a ansiedade em subir logo de cargo, sobre a realidade do mercado, os motivos pelos quais, em alguns momentos, a empresa nem mesmo pode nos dar uma promoção e até mesmo sobre quando o nosso ego nos coloca em situações para as quais não temos preparo. Mas também existe o momento em que tudo está ao nosso favor e mesmo assim nunca chega o nosso dia.

Como comentado anteriormente, todos nós desejamos reconhecimento e crescimento, mas existem muitas variáveis que podem nos segurar no mesmo cargo. Porém nós aprendemos a refletir sobre o mercado e entendemos essas variáveis por parte da empresa. O problema acontece quando a nossa gestão não trabalha com honestidade e transparência e fica sempre nos prometendo que, no próximo ciclo, será a nossa vez.

Caso essa empresa não nos apresente nenhuma vantagem, talvez seja a hora de procurar uma nova oportunidade no mercado. Eu tenho uma regra pessoal para garantir que esteja tendo alguma vantagem para o longo prazo: meu emprego atual precisa me dar dinheiro ou aprendizados, se entregar os dois, é melhor ainda. Se você só tem uma promessa de aumento ou promoção que nunca se realiza, talvez seja hora de repensar este lugar. Aqui eu estou assumindo que você, que já leu tudo o que eu disse anteriormente, já consegue identificar se tem preparo para assumir um novo cargo ou não. Mas existem ainda motivos mais complexos pelos quais a sua promoção não está vindo.

No último censo do IBGE, foi contemplado que uma brasileira recebe, em média, 21% a menos do que um homem no **mesmo cargo, com a mesma formação e tempo de experiência**. Em uma pesquisa da Catho, plataforma de vagas de trabalho, descobriram que, mesmo **com maior grau de escolaridade**, as

mulheres ganham menos do que homens. Isso só nos evidencia que ainda temos muita desigualdade de gênero no mercado de trabalho. 30% das mulheres possuem nível superior e pós-graduação, enquanto homens com este currículo são 24%. Mesmo assim, eles podem chegar a ganhar até 52% a mais do que elas exercendo uma mesma função.

Segundo o Ministério Público do Trabalho, pessoas negras enfrentam dificuldade na progressão de carreira, na igualdade de salários e são mais vulneráveis ao assédio moral no ambiente de trabalho. A renda média de trabalhadores brancos é 75,7% maior do que a de pretos, segundo o IBGE. De acordo com o levantamento, os brancos ganham R$ 3.099 em média. Esse valor é 75,7% maior do que o registrado entre os pretos, que é de R$ 1.764, e também supera em 70,8% a renda média de R$ 1.814 dos trabalhadores pardos.

Uma pesquisa publicada no periódico científico *Journal of Population Economics* revelou que, em média, pessoas heterossexuais recebem salários 6,8% maiores do que pessoas LGBTQIAP+. O estudo foi realizado em 24 países, entre eles Estados Unidos, Austrália e algumas nações da União Europeia. O levantamento indicou que a diferença entre os pagamentos varia de acordo com a orientação sexual. Homens bissexuais ganham 10,3% menos do que heterossexuais, enquanto mulheres bissexuais recebem 5,1% menos do que mulheres heterossexuais.

As pessoas com deficiência apresentaram, em 2019, taxas de participação no mercado de trabalho (28,3%) e de formalização (34,3%) muito menores do que as das pessoas sem tal condição (66,3% e 50,9%, respectivamente), sendo a desocupação observada entre pessoas com deficiência (10,3%) maior do que a verificada entre as pessoas sem deficiência (9,0%). Pessoas com deficiência são tão esquecidas pela nossa sociedade que, para os dados anteriores sobre pesquisas relacionadas a mulheres, pessoas negras e LGBTQIAP+, eu encontrei também matérias em agências de notícia e sites de notícia populares, enquanto os dados que eu estou

trazendo agora foram extraídos direto do site do IBGE, sem nenhuma facilitação do conteúdo para o público geral.

O que estou tentando dizer é que a desigualdade e o preconceito nos afetam tanto no mercado de trabalho quanto o humor dos investidores. Talvez seja hora de olhar ao redor e perceber quantas pessoas iguais a você estão em cargos de liderança, gestão ou superiores; quantas pessoas do seu recorte social estão recebendo aumento ou promoções durante os ciclos de avaliação. Esse exercício pode responder muitas questões internas que só você sabe que tem.

Caso você perceba que este não é mais o lugar onde você deveria estar, então precisamos conversar sobre os processos seletivos para pessoas com histórico de desigualdade social. Segundo o Informe de Percepção de Gênero, divulgado pelo LinkedIn, mulheres aplicam para 20% menos vagas do que os homens. Elas sentem que precisam cumprir 100% dos requisitos das vagas de emprego, enquanto a maior parte dos homens arrisca com apenas 60%.

É hora de mudar isso. Se você sente que já está na hora da sua promoção, você pode seguir os passos dos próximos capítulos e buscar isso. Mas caso perceba que este lugar não vai lhe dar reconhecimento devido a problemas estruturais, é hora de aplicar para vagas em empresas que possam lhe trazer alguma daquelas vantagens que eu comentei anteriormente e/ou um ambiente mais seguro para o seu crescimento e felicidade profissional.

Conclusão

Eu não estou aqui para jogar um balde de areia em seus sonhos, muito menos limitar o seu potencial como pessoa e profissional. As experiências que comentei neste capítulo devem servir como reflexão para colocarmos os pés no chão e a ansiedade não tomar conta de nossas vidas profissionais.

Nem sempre aquilo em que acreditamos é uma verdade. E no ambiente profissional em engenharia de software, isso acontece o tempo todo. Porém, quando é só no código, é mais fácil de corrigir do que quando é dentro de nossas cabeças. Por isso eu deixei esses conselhos antes de seguirmos para o plano de carreira em si.

Nos próximos capítulos, seguiremos com a parte prática de como é atuar como especialista. Vamos conversar sobre promoções, entrevistas de emprego e mais.

Referências

- *Efeito Dunning–Kruger* (WIKIPÉDIA, c2022): https://bit.ly/dunning–kruger

- *Síndrome do impostor* (WIKIPÉDIA, c2023): https://bit.ly/-sindrome-impostor

- *Por que mulheres se candidatam a menos vagas de emprego que homens* (MARTINEZ, 2020): https://bit.ly/tab-uol-estudo-linkedin

- *Estudo: Heterossexuais ainda ganham mais do que LGBTs* (BRITO, 2021): https://bit.ly/lgbts-ganham-menos

- *Pessoas com Deficiência e as Desigualdades Sociais no Brasil* (IBGE, 2022): https://bit.ly/pessoas-com-deficiencia-ibge

- *IBGE: renda média de trabalhador branco é 75,7% maior que de pretos* (RODRIGUES, 2022): https://bit.ly/renda-pessoas-negras

- *Negros enfrentam mais dificuldades que brancos no mercado de trabalho, diz MPT* (BOEHM, 2017): https://bit.ly/mpt-negros-dificuldade-mercado

- *Mulheres têm rendimento 21% inferior ao dos homens, mostra pesquisa* (BOCCHINI, 2023): https://bit.ly/rendimento-mulheres

- *Mulheres no mercado de trabalho: panorama da década* (CARMO, 2023): https://bit.ly/catho-pesquisa-mulheres

Capítulo 7
A consolidação como sênior

No capítulo anterior, refletimos sobre como a ansiedade nos faz desejar sempre pular para a próxima etapa, muitas vezes sem nem mesmo ter o preparo para isso, seja formação, experiência ou segurança psicológica. Por isso, neste momento, vamos entender alguns pontos que podem nos ajudar a buscar essa consolidação como sênior antes de enfrentar o cargo de especialista ou mesmo de começar a se preocupar com isso.

Vamos imaginar essa consolidação como uma trilha de alguns passos:

1. Alcançar um bom nível técnico;
2. Dominar o não técnico;
3. Gerar impacto no time;
4. Ser exemplo.

7.1 Alcançar um bom nível técnico

Nosso trabalho como pessoas desenvolvedoras é técnico. A maior parte do nosso trabalho é focada em engenharia de software, não em usabilidade ou finanças. Normalmente, para esses outros tópicos, temos outras pessoas especialistas nesses temas que cuidam disso, como UX designers, que cuidam da usabilidade, e pessoas do setor financeiro, que cuidam da liberação de dinheiro para os times, por exemplo.

Nós precisamos escrever código para entregar as novas funcionalidades de um software, corrigir bugs, gerar valor financeiro para a empresa ou impacto social na organização sem fins lucrativos. Não somente escrever código, mas monitorar o software,

fazer testes de segurança, testes de carga etc. Temos que manter as coisas funcionando da melhor maneira possível.

Como pessoas seniores, normalmente não somos nós que vamos ter que escrever 100% do código. Às vezes, nem 50%. Temos muito mais impacto quando automatizamos o trabalho de outras pessoas desenvolvedoras, quando fazemos o levantamento de requisitos, quebramos histórias em tarefas menores e mais impactantes para o cliente ou estruturamos algo para o futuro que uma pessoa com menos experiência não conseguiria fazer com eficiência.

Para conseguir ter segurança em nosso nível profissional, precisamos garantir o conhecimento do ciclo de vida de um software. Será necessário ter um bom conhecimento em nossas ferramentas, como a linguagem de programação, a infraestrutura que roda nosso software, o monitoramento da saúde do software etc. Precisamos garantir que realmente entendemos tudo o que envolve nosso software.

Como seniores, seremos as pessoas que serão consultadas para garantir que algo é possível de ser feito, como a implementação de uma nova ideia do time de produtos, quanto tempo isso demoraria para ser feito e até mesmo se isso deveria estar no nosso software ou em outro produto dentro da empresa/organização. Não tem como garantir que tomaremos boas decisões se não temos o conhecimento técnico aprofundado o suficiente para enxergar coisas que outras pessoas não enxergam.

Um exemplo de atuação técnica é na revisão de *pull requests*. As pessoas mais juniores podem colocar em produção códigos que, depois, serão muito difíceis de dar manutenção ou de entender caso aconteça um bug. Nós precisamos ter conhecimento profundo da nossa linguagem de programação, padrões de projetos e algoritmos para evitar esses problemas.

Outro exemplo é no monitoramento e debug dos sistemas. Uma pessoa menos experiente pode encontrar um problema de

performance em nossa aplicação e não conseguir identificar corretamente onde está o erro, se é algo relacionado ao código-fonte, se é problema de latência de rede, se falta capacidade de processamento no servidor etc. São muitos os cenários, e nós só conseguimos pensar em todos esses cenários se realmente temos o conhecimento necessário para isso.

Se você ainda não se sente confortável com o conhecimento técnico que possui no ciclo de vida de um software, talvez ainda não seja a hora de assumir o próximo desafio, porém isso não é uma verdade absoluta. Algumas pessoas só conseguem aprender quando estão no meio da prova de fogo. Falta-lhes motivação para estudar. Não é só ter curiosidade sobre um tema, existem motivações internas e externas para o que fazemos no nosso dia a dia. Existem pessoas que, internamente, conseguem olhar um tema e se aprofundar nele, mesmo que não utilizem isso em seu dia a dia, e ficam esperando o dia em que vão precisar disso. Outras pessoas precisam de um estímulo externo para conseguir essa motivação, como um problema que ela ainda não sabe resolver, um feedback da gestão ou algo similar. Será necessário entender como você se motiva para, enfim, conseguir esse aprofundamento técnico.

Se você é uma pessoa com motivação interna, uma sugestão é conversar com outras pessoas seniores, pessoas que estão acima de você na organização e também buscar conhecer a imagem do todo de como os sistemas da organização são construídos. Isso vai ajudar você a montar uma lista de tópicos sobre o que você sabe mais ou menos, e os que você domina menos precisam de foco para serem reforçados antes de dar o próximo passo.

Se você é uma pessoa que precisa de estímulo externo, uma sugestão é se comprometer com outras pessoas. Muitas vezes só estar em um projeto não traz desafio técnico suficiente para que nós precisemos buscar esse conhecimento profundo, por isso podemos alinhar com a gestão alguma meta que nos ajude a focar. Por exemplo, se comprometer a subir a cobertura de testes pode ajudar a forçar aprender mais sobre qualidade de software ou colocar como

meta uma palestra técnica na organização ou para a comunidade de software.

Eu sou uma pessoa que precisa de estímulos externos para estudar tópicos relacionados a trabalho. Adoro estudar engenharia de software, por isso eu já li livros que só nos são apresentados em boas universidades, mesmo que eu não tenha me graduado em uma dessas. Porém eu trabalho como desenvolvedor web, e existem diversos tópicos que não me são interessantes, como HTML ou CSS. Para melhorar nesses pontos, eu criei um blog onde eu guardo minhas anotações dos estudos. Além disso, busco estar sempre conectado com a comunidade de desenvolvimento web observando esses tópicos que eu não tenho tanto interesse e tentando ajudar as pessoas a resolverem suas questões em fóruns ou *meetups*. Isso faz com que eu também tenha que estudar para conseguir me manter ajudando as pessoas, que é algo que me motiva bastante. Ou seja, eu descobri que o que me motiva internamente é ajudar outras pessoas, então eu junto o útil (estudar coisas que eu não tenho interesse) ao agradável (compartilhar essas coisas com pessoas que precisam).

Entretanto, mesmo com tudo isso em nossas mentes, não existe um roadmap aberto na internet sobre **quando** atingimos de fato essa consolidação. Por isso é importante focar em nosso contexto.

Independente de como você se motiva, procure entender o todo dentro da empresa: quais são as tecnologias usadas, as práticas (convenções, metodologias etc.) de engenharia, infraestrutura, observabilidade e qualquer outro tema técnico que seja importante para a organização, e foque nisso. Se você tentar focar em **tudo** o que existe em engenharia de software, você nunca alcançará essa consolidação.

Nossa especialidade como pessoas desenvolvedoras de software é **aprender**. Descubra como isso funciona melhor para você e esteja sempre aprendendo, mas não crie barreiras mentais que possam travar sua evolução profissional. Achar que precisa aprender 5 ou 10

linguagens de programação, dominar todos os comandos úteis e inúteis do Linux ou Git para o seu dia a dia, escrever um compilador, sendo que você trabalha com integração entre serviços, ou qualquer outra premissa inalcançável para ser sênior é perda de tempo.

Entenda o todo dentro do seu contexto e foque nisso até sentir que entendeu e consegue ter utilidade resolvendo problemas dentro deste contexto. Se você trabalha com desenvolvimento web, entenda os protocolos de rede, como os servidores funcionam, como os navegadores funcionam, domine sua linguagem de programação. Não faz sentido dominar tudo relacionado às bibliotecas JavaScript se você trabalha com desenvolvimento de embarcados usando a linguagem C. Tudo depende de contexto.

7.2 Dominar o não técnico

Nem só de linguagens de programação viverá a pessoa sênior. O nosso dia a dia profissional é envolto de trabalho em equipe, reuniões com clientes, programação em pares com colegas mais ou menos experientes e muita interação humana. Por isso, não adianta somente ser a pessoa mais técnica do time, talvez isso não garanta a oportunidade de evolução profissional.

Como eu já disse em outros capítulos até aqui, o papel de uma pessoa especialista é muito estratégico e não tanto de interação unicamente com código. Por isso é extremamente importante que busquemos vivência além disso.

Aqui nós temos uma grande questão: algumas pessoas se envolvem com o trabalho com tecnologia porque precisam de menos interação com outras. Infelizmente (para mim) isso não dura a vida inteira. Como seniores, será extremamente necessário começar a compartilhar nosso conhecimento e ajudar a organização a difundir seus padrões e convenções. Caso nós não tenhamos um bom trato

social, podemos acabar sendo aquela pessoa que desmotiva o time ao invés de ser a que puxa as pessoas para cima e as auxilia a serem melhores.

Eu não tinha uma boa relação com pessoas, por isso eu demorei para ter um comportamento construtivo no dia a dia de trabalho. A única maneira que encontrei de melhorar neste ponto foi através do processo terapêutico. Nossa liderança e gestão não consegue nos ajudar em todos os pontos de nossa vida. Muitas vezes recebemos sugestões da liderança que são extremamente impactantes e positivas para o nosso futuro profissional, mas, quando se trata de autoestima, traumas pessoais e outros diversos pontos que impactam nosso dia a dia, não tem como a gestão nos ajudar. Estou comentando isso pensando em quem também não consegue destravar com pessoas. Para o caso de quem já "se dá bem com todo mundo", a gestão consegue direcionar muito bem.

Antes de continuar, precisamos refletir sobre algo aqui: **SUA LIDERANÇA OU GESTÃO NÃO É SEU/SUA PSICÓLOGO/A.**

Essa pessoa pode ou não ter a capacidade de lhe ajudar em diversos cenários, mas você não pode descarregar todos os seus traumas em cima dela e acreditar que receberá a resposta para os problemas do universo. Lembre-se sempre de que você está conversando com outro ser humano e profissional que tem suas próprias questões e cobranças da empresa. Tirando quem não trabalha para sobreviver, estamos todos no mesmo barco.

Também não aceite sugestões baseadas em diagnóstico de pessoas nesses cargos. Algumas pessoas acreditam que, por serem boas com as habilidades pessoais, elas também conseguem enxergar dentro de nós coisas que não enxergamos. Não, elas não conseguem. Elas não sabem quando você está em depressão; também não conseguem dizer se você tem ansiedade generalizada. Caso alguém venha dizer isso para você, desconsidere. Só quem pode fazer isso é o/a médico/a psiquiatra.

Para este ponto, eu prefiro citar referências que me ajudaram a mudar minha forma de comunicação e convivência, que são os livros:

- *Conversas difíceis: Como discutir o que é mais importante*, por Bruce Patton, Douglas Stone e Sheila Heen (Sextante, 2021);
- *Obrigado pelo feedback*, por Sheila Heen e Douglas Stone, (Portfolio-Penguin, 2016);
- *Comunicação não violenta*, por Marshall B. Rosenberg, (Ágora, 2021).

Além dessas leituras extras, recomendo fortemente que busquemos a terapia para nos entendermos melhor e descobrir por que temos determinadas atitudes para algumas situações. Só isso vai nos preparar para termos uma boa vida laboral, além de nos garantir

segurança psicológica para passar mais alguns anos trabalhando com cada vez mais responsabilidades e estresse.

7.3 Gerar impacto no time

Uma vez dominado o técnico e o não técnico, agora conseguimos começar a pensar no impacto no time. Seria muito difícil pensar nisso antes de ter um bom conhecimento técnico e boa relação com as pessoas, pois, a partir de agora, temos que começar a compartilhar o nosso conhecimento, além de estimular as convenções da empresa.

Existem muitas formas de impacto em um time de engenharia, e a principal é o compartilhamento de conhecimento. Através dessa prática, nós conseguimos equalizar as habilidades do time. Quanto mais conhecimento todos possuem, melhor o trabalho que vamos fazer juntos — e juntos conseguimos entregar muito mais do que individualmente.

Em diversos momentos deixamos de compartilhar o que sabemos, como em reuniões de planejamento, onde poderíamos comentar mais sobre algo que imaginamos estar indo para o caminho errado, ou em *pull request* que precisam de certa atenção e deixamos passar. Para que isso realmente aconteça, precisamos mudar a nossa postura para a de alguém que está na disposição de ensinar.

Pair programming

Trabalhar em pares é uma maneira interessante de ensinar algo para alguém. Através dessa prática, podemos convidar uma pessoa com menos experiência para nos acompanhar em uma tarefa mais complexa ou podemos auxiliar em tarefas mais fáceis ensinando como nós fazemos as coisas.

Pessoalmente, eu não gosto de como algumas empresas utilizam essa prática, estimulando que todos trabalhem sempre em pares. Isso é extremamente controverso para pessoas neurodivergentes. Eu mesmo não consigo manter meu foco por muito tempo em um mesmo ponto ou no que alguém está falando. Se eu tiver um gatilho de ansiedade durante o *pair programming*, eu posso acabar atravessando o raciocínio da outra pessoa muitas vezes durante o pareamento.

Eu prefiro o *pair programming* enquanto ferramenta de compartilhamento de conhecimento e trabalho em equipe, quando precisamos resolver algo muito complexo que as pessoas não conseguem resolver sem auxílio, ou mesmo quando as pessoas gostam e conseguem trabalhar assim por muito tempo, mas nunca como uma obrigação.

A convenção do que é o *pair programming* nos estimula a ter uma pessoa escrevendo o código e outra pessoa guiando a forma como a tarefa deve ser realizada. Essas pessoas são conhecidas como piloto e copiloto, e é como em uma prova de rally, em que o piloto dirige o carro e o copiloto diz aonde ir. Durante a evolução do desenvolvimento, as pessoas trocam de posição.

Eu também gosto de utilizar uma técnica que aprendi com meu colega de trabalho, na época, meu sênior, José Teles. Nós fazíamos o que ele chamava de "***pairmodoro***", que é exatamente o que ela diz em seu nome: a mistura de *pair programming* com a técnica *pomodoro*. Portanto, nós fazíamos 25 minutos codificando, focados, realizávamos uma pausa de 5 minutos e, na volta ao trabalho, trocávamos de posição de piloto para copiloto e vice-versa.

O *pair programming* é uma ótima ferramenta que eu vou citar mais vezes durante a leitura deste capítulo, pois é uma ótima maneira de inspirar outras pessoas através do exemplo.

Compartilhando informação

Precisamos criar espaços em nossa agenda para termos foco nos *code reviews* e poder ter atenção no que estamos fazendo. Dessa forma, conseguimos encontrar pontos de melhorias no código das outras pessoas e comentar com maior riqueza de detalhes e fontes de informação. Esse espaço na agenda também será útil para os momentos em que as pessoas não entendem muito bem o que comentamos. Podemos convidar essa pessoa para um *pair programming* ou mesmo para um bate-papo rápido explicando nossos pontos.

É extremamente importante lembrar que as pessoas têm níveis de conhecimento diferentes, assim como fontes de informação mais ou menos eficientes. Por isso, podemos sempre colocar links para ajudar a reforçar o que estamos comentando ou mesmo exemplos de código de como algo deveria ser.

Eu gosto muito de utilizar o site *refactoring.guru* quando preciso comentar sobre padrões de projetos, por exemplo. Mas podemos nos apoiar em qualquer tipo de recurso que nos ajude a mostrar algo importante para as pessoas do time, como canais no YouTube, blogs etc.

Tenha sempre bons links em sua caixinha de ferramentas. Eu guardo tudo no meu blog para facilitar minha busca: woliveiras.com.br/my-precious-links.

Reuniões

Nas reuniões técnicas ou de produto, é muito comum que pessoas menos experientes se esqueçam de algum detalhe nas tarefas, seja algo relacionado a código ou ao planejamento. Este é outro momento em que podemos estimular o raciocínio das pessoas e tentar trazer esses pontos. Com o tempo, elas vão levantar essas questões por conta própria, gerando mais eficiência no longo prazo.

É muito difícil manter a atenção em reuniões quando não temos interesse nos tópicos apresentados — e, na maioria das vezes, o

nosso dia a dia é recheado dessas reuniões. É importante que tenhamos algum artifício que nos ajude a focar, pois, como seniores, temos grande responsabilidade nesses momentos. Eu utilizo o recurso de anotar pontos-chaves. Independente da reunião, eu utilizo meu caderninho para ir anotando frases-chaves e pontos que parecem ser importantes durante o bate-papo. Isso faz com que eu preste mais atenção nesses momentos não muito legais. Durante a reunião ou no final da conversa, podemos revisar esses pontos com quem estava apresentando.

Utilizar esses recursos para focar durante as reuniões vai nos ajudar a perceber quando as pessoas com menos senioridade estão deixando passar alguma coisa. Isso nos ajuda a antecipar problemas.

Tech talks

Algo que adoro como profissional de tecnologia é o fato de que temos muitas maneiras de compartilhar conhecimento em nossa área, seja através de um blog, canal em plataformas de vídeo, palestras; dentro das organizações, também temos as *tech talks*.

Caso você não tenha contato com essa prática, normalmente as *tech talks* são palestras rápidas nas quais as pessoas da companhia compartilham algo que dominam, têm curiosidade ou gostam bastante. Pode ser algo relacionado ao que é usado na empresa ou mesmo uma coisa que você tem vontade de um dia utilizar dentro da organização.

Uma *tech talk* pode ser utilizada para compartilhar boas práticas que faltam para o nosso time, mas também pode ser uma ferramenta para nos manter em busca por conhecimento. Podemos colocar uma pequena palestra como meta para depois de aprofundar em algum tema.

Dojos

Assim como as *tech talks*, dojos não acontecem em todas as organizações, mas são ótimas maneiras de estimular o trabalho em equipe, a comunicação eficiente e o conhecimento técnico.

Dojos são eventos de código onde nos juntamos em frente a um único computador conectado a um projetor e duas pessoas utilizam o *pair programming* para resolver um problema de código. O que é interessante nesse modelo é que precisamos trabalhar com desenvolvimento orientado a testes, onde escrevemos primeiro os testes de unidade e em seguida codificamos os módulos em si. As duas pessoas que estão em par podem ou não trocar com a plateia. Mas, em geral, todos estão ali para se ajudar, então a plateia também dá suas sugestões do que fazer e como fazer, sempre seguindo pequenos passos até a realização da tarefa.

Dojos são extremamente úteis para aprender algo novo, seja para pessoas seniores ou juniores: para juniores, aprender a escrever bons testes de unidade e a trabalhar em *baby steps* ("passos de bebê", ou pequenos passos); para seniores, trabalhar a comunicação e poder utilizar esses eventos para treinar/aprender alguma tecnologia nova.

Documentações

Documentar é, de longe, a prática mais deixada de lado por pessoas desenvolvedoras de software. É muito difícil engajar em fazer algo que não tem um resultado eminente, e documentar é justamente esse caso. Nós só conseguimos ver o sucesso da documentação com o longo prazo ou mesmo quando utilizamos código *open source*, mas não é a tarefa que gera mais destaque em uma reunião de resultados.

Acredito que o maior desafio de documentar é o fato de que estamos o tempo todo correndo atrás de entrega de novas funcionalidades devido a necessidades de negócio. Mas nós, como seniores, precisamos defender a criação e manutenção desses documentos, pois vai facilitar o nosso dia a dia, assim como vai

melhorar a experiência de pessoas que entram no time ou de pessoas com menos experiência que não conseguem entender por que as coisas são como são dentro da nossa organização.

Busque tempo para documentar, assim como para fazer bons *code reviews*. Isso vai gerar eficiência no time e facilitar a comunicação dentro da empresa. Em vez de você precisar parar o que está fazendo para explicar algo o tempo todo, pode simplesmente mandar a documentação e responder somente as dúvidas que as pessoas tiverem. Depois você pode atualizar a documentação com essas novas dúvidas a fim de evitar esse trabalho novamente no futuro. Documentar faz parte da busca por eficiência no nosso dia a dia.

7.4 Ser exemplo

Compartilhar conhecimento, estimular a evolução técnica e profissional das pessoas, participar de reuniões e outros eventos aos quais não temos tanto interesse: o trabalho da pessoa sênior acaba nos requisitando muito preparo psicológico e um bom comportamento em equipe.

É importante entendermos que ser sênior não é ser dono(a) da verdade ou ser sempre quem dá a última palavra em alguma discussão. O mais importante do nosso dia a dia é ser exemplo para as outras pessoas.

Quando estamos em um time onde todos se preocupam com qualidade, com o tempo, passamos a nos preocupar com qualidade. Quando estamos em um time onde ninguém liga para isso, logo também deixamos de se importar.

A pessoa sênior possui uma certa admiração intrínseca do time. Precisamos utilizar essa admiração de modo que encaminhemos as pessoas para um bom caminho. Se estamos demonstrando

preocupação com as coisas certas dentro da organização, as demais pessoas passam a também valorizar isso. Não é necessário forçar ninguém a fazer algo. O tempo faz com que seu bom exemplo gere impacto.

Além disso, você precisa seguir as convenções e boas práticas, mesmo que você não concorde com elas, assim como estar em reuniões chatas e buscar pontualidade no seu dia a dia. Você precisa realmente ser o exemplo de profissional que as outras pessoas devem buscar.

Claro que também existem pessoas que não querem seguir o que é certo, independente de como nós tentamos ter empatia. Para esses casos, existe a gestão. Você não precisa esquentar sua cabeça com problemas comportamentais das pessoas. Caso encontre resistência de uma única pessoa, é importante levar este caso para a gestão e pensarem juntos se precisam mudar a abordagem ou mesmo se a gestão precisa intervir de algum modo. As pessoas são plurais, elas podem não estar entendendo o que queremos, podem ter problemas pessoais ou só ter uma personalidade resistente.

Para todos os casos, não adianta levantar a voz ou ter qualquer outro comportamento autoritário, esse tipo de comportamento destrói o time com o tempo ao invés de melhorá-lo.

Conclusão

Muito do nosso trabalho para nos consolidarmos como seniores está em buscar conhecimento, compartilhar e buscar eficiência no dia a dia. Mas também é importante entender que isso também vem com o tempo. Não existe um tempo certo para se tornar sênior, assim como não existe uma métrica de quando temos o conhecimento consolidado, por isso precisamos aprender a entender nosso contexto.

Sobre compartilhar conhecimento, não se esqueça da parte de liderança pelo exemplo. Não conseguimos estimular as outras

pessoas através da ignorância ou palavras que reforçam o sentimento de superioridade por termos mais experiência. É somente através do bom exemplo que as pessoas sentirão vontade de ser iguais a nós.

Capítulo 8
Colocando em prática

Nós já aprendemos muito até aqui. Estamos mais conscientes sobre como o mercado funciona, como os cargos se relacionam dentro da organização **e com** a organização, e já temos informação sobrando sobre o universo da pessoa especialista em engenharia de software.

Talvez você já até tenha o seu plano de como se consolidar como sênior e se mover para especialista, ainda mais se você tiver uma boa liderança e gestão próxima, se já conversaram sobre isso anteriormente, já traçaram as metas e próximos passos para você seguir. Mas, caso essa não seja a sua realidade, não se preocupe. Neste capítulo, vamos aprender a colocar tudo isso em prática.

Vamos por etapas, sempre buscando aquele equilíbrio do qual eu falo desde o começo. Começaremos pelo momento antes da promoção, passaremos pelo instante em que informamos à nossa gestão de que desejamos seguir a trilha técnica e, em seguida, vamos assumir o dia a dia de trabalho nesta carreira, aprendendo novas maneiras de atuar, posicionamento estratégico em um time/uma organização, gestão do nosso tempo e outros conselhos práticos do dia a dia como especialista.

8.1 Antes da promoção

Um motivo de insatisfação que eu tive durante minha carreira profissional é o fato de que, às vezes, é possível ver quando as pessoas ao redor já estão prontas para assumir um próximo nível, mas, por algum motivo, não recebem esse reconhecimento da sua gestão.

Isso foi algo que me afetou profundamente durante muito tempo, até o dia em que eu questionei um gestor de outra equipe sobre o porquê de uma pessoa não ser promovida no ciclo de avaliação. Foi ele quem começou a abrir minha visão sobre como o mercado de trabalho e a parte financeira de uma empresa funcionava, mas o que me surpreendeu foi uma frase que ele usou para comentar justamente sobre a pessoa que eu estava citando:

> Existem dores no nosso corpo que não geram sintomas visuais pelos quais outras pessoas conseguem perceber que você precisa de ajuda. Nesse caso, você precisa **DIZER ATIVAMENTE** que precisa de socorro.

O que ele quis dizer com isso foi que ele também sabia que aquela pessoa estava preparada para uma promoção. Ele queria promovê-la e estava fortemente disposto a reunir dados para defendê-la em um próximo ciclo de avaliação. Mas em todas as vezes ele que tocava nesse assunto, a pessoa não exibia nenhum interesse em uma promoção ou em assumir mais responsabilidade, o que é extremamente comum em qualquer área. Existem pessoas que querem assumir mais responsabilidade e outras que desejam continuar onde estão, se sentem bem com isso, são produtivas e geram lucros para as empresas. Não existe nada demais nessa situação, e eu aprendi isso nessa conversa. Mas também aprendi que não adianta só termos a preparação necessária para o próximo cargo, precisamos demonstrar para a nossa gestão que temos esse desejo.

Então, o primeiro passo para uma promoção é demonstrar interesse para a nossa liderança direta ou gestão. Mas não adianta chegar dando um ultimato, no qual, em vez de informar que deseja uma promoção, você **exige** isso. Lembre-se dos pontos que citei no capítulo anterior: existem várias coisas acontecendo na organização que podem impactar de forma positiva ou negativa sua possibilidade de promoção.

Antes de pedir uma promoção, é importante saber se estamos cumprindo com os objetivos que a empresa espera de uma pessoa em nossa posição. Por isso, é ideal que pergunte à sua gestão: **"Como estou performando? Estou abaixo, acima ou cumprindo as expectativas?"**

Existem várias atitudes que precisamos tomar para conquistar nosso objetivo, por isso é importante alinhar com quem está acima de nós na hierarquia, senão podemos acabar executando ações que não ajudam em nada. Neste livro (e até mesmo na internet) encontramos diversas informações sobre o que fazer para conseguir uma promoção, mas só quem sabe o que é importante para a sua organização no momento atual é quem está acima de você. Por isso, essa conversa com a gestão é mais importante do que qualquer atitude proativa de nossa parte.

Com a resposta para essa pergunta, conseguimos seguir para o próximo passo. Se estamos indo mal, precisamos alinhar novos objetivos com a nossa gestão a fim de melhorar esses indicadores. Se estamos cumprindo ou estamos acima das expectativas, podemos então informar que temos interesse em subir de cargo. Podemos utilizar outras perguntas-chaves, como **"o que eu preciso fazer para conseguir uma promoção no próximo ciclo?"** ou **"quais as chances de que eu receba uma promoção no próximo ciclo de avaliação?"**. Essas perguntas vão abrir espaço para conversar sobre a existência ou não da possibilidade de subir de nível.

Algo comum de acontecer nesse momento é nossa gestão nos solicitar que tomemos algumas atitudes que não estávamos tendo anteriormente. Vou citar aqui alguns exemplos do que eu já recebi de feedback para exemplificar o que pode acontecer com você:

- Você precisa começar a puxar a régua do time em relação à qualidade e segurança de software, ser mais crítico nos *code reviews* e não deixar passar código sem qualidade;

- Você precisa assumir a liderança por este projeto que estava engavetado e agora será seu; conecte-se com outros times, outras lideranças e faça o que for preciso para que este projeto saia do papel;
- Você precisa buscar uma pessoa para mentorar você a fim de melhorar seus conhecimentos técnicos antes de assumir o próximo nível, pois ainda falta conhecimento em arquitetura de software e padrões de projetos, coisas que serão cobradas de você no próximo cargo.

Com base nos comentários que recebemos, agora podemos combinar alguns objetivos alcançáveis e mensuráveis com a nossa gestão. Assim, teremos sempre a realidade sobre se estamos conseguindo realizar o que nos foi proposto durante as próximas conversas.

Com estes objetivos em mãos, podemos ter foco para que nossa gestão leve esses resultados em nossa defesa na hora de pleitear uma promoção. Para conseguirmos nossa promoção, precisaremos alinhar com a nossa gestão que temos interesse no próximo cargo, traçar os objetivos para o ciclo de avaliações atual, a fim de conseguir gerar bons resultados que justifiquem o reconhecimento, e combinar um ciclo de feedbacks, para acompanhar se estamos conseguindo realizar as tarefas deste objetivo ou se precisamos de ajuda ou de mudar alguma coisa nos planos.

Mas sempre se lembre de que qualquer combinado aqui não será garantia de uma promoção e, mesmo que essa promoção não venha, essa é a postura que você deve assumir daqui para a frente, sempre buscando mais responsabilidades, sempre tentando trabalhar no que gera impacto e não em tarefas simples que qualquer outra pessoa desenvolvedora do seu time pode fazer. Isso não só vai ajudar uma possível evolução no futuro, mas na sua performance profissional como um todo.

8.2 Assumindo a postura de especialista

Uma das partes mais difíceis da promoção de sênior para especialista é a mudança que precisamos fazer em nossas mentes. Não vamos mais atuar como fazíamos antes. Assumiremos uma postura muito mais estratégica e pouco mão na massa, se comparada com os cargos até chegar aqui.

A partir do momento em que assumimos o papel de especialista, seja como liderança técnica ou dentro dos diferentes níveis de staff+ levantados nos capítulos 3 e 4, o nosso trabalho não será mais somente abrir *pull requests* em nossos sistemas, participar de algumas reuniões e ajudar o nosso time. O impacto do que fazemos agora pode mudar a direção da empresa e, por isso, as coisas parecem caminhar mais devagar, além de tudo parecer muito mais crítico e você precisar de muito mais argumentação para conseguir uma mudança.

Modificar um código, executar os testes, abrir um *pull request*, mesclar o código novo com o código em produção e publicar a nova versão: esse é o caminho que nós tanto adoramos em engenharia de software.

Mas, como especialista, talvez o seu trabalho seja escrever a documentação de arquitetura que possibilita a construção desse sistema, ou estar na reunião com as lideranças técnicas a fim de garantir que todos estão seguindo as diretrizes da empresa, ou mesmo escrever código, mas algo que será usado internamente pelas próprias pessoas desenvolvedoras para gerar *boilerplate* para iniciar projetos, um trabalho muito menos emocionante do que validar código novo em produção.

Isso nos gera um choque, pois antes nós colocávamos o sistema no ar, agora nós participamos do planejamento do sistema, mas não necessariamente escrevemos o código que vai levantar o produto. Será que as pessoas que estão desenvolvendo o sistema o estão fazendo como sugerimos? Será que o que nós planejamos cumpriu o

objetivo do time? Esses e outros vários "serás" surgirão em nossa cabeça a partir de agora.

Para as coisas começarem a fazer sentido em nossa cabeça, precisamos aprender a mensurar **impacto versus produção**.

Quando falamos de produção, estamos realmente pensando em quantidade de linhas de código, quantidade de bugs resolvidos, *deploys* realizados e outras métricas relacionadas à construção e manutenção dos sistemas de uma organização.

Pensar em impacto é diferente: estamos falando agora sobre como a modelagem de uma arquitetura de software pode travar a produção de mais código no futuro, por exemplo.

Pensando em baixo nível e visando os produtos de uma organização, muitas vezes escrevemos (muito) código para um sistema e este produto, depois de lançado, não gera nenhum lucro, levando a algum prejuízo ou até precisando ser descontinuado para estancar o escoamento de dinheiro que um produto não lucrativo pode causar. Neste cenário, foram vários meses de trabalho em um sistema que não gerou impacto para a organização.

Como especialistas, o nosso trabalho atinge vários times e vários produtos. Pode ser que um deles seja descontinuado, mas, independente disso, o seu código, o seu planejamento, os componentes de arquitetura que desenvolveu ou qualquer iniciativa gerará valor para a empresa no longo prazo.

Muitas vezes, ao entrar em uma nova empresa, você se depara com decisões arquiteturais que foram tomadas há muito tempo por uma pessoa especialista. Seja uma boa decisão ou uma péssima, nosso trabalho permeia por muitos anos a organização, diferente do código que uma pessoa júnior, pleno ou até mesmo uma sênior escreveu e pode ser jogado fora daqui a algum tempo.

Perceba: impacto nem sempre é visto hoje. Talvez você trabalhe em uma tarefa organizacional que impacta tantos sistemas e tantas

pessoas que esse projeto vai demorar alguns anos. Pode ser que você nem veja a conclusão desse projeto, por trocar de emprego ou mesmo por deixar o trabalho rolar e ir fazer alguma outra coisa.

Essa é a diferença entre impacto e produção. Podemos produzir muito rápido, escrever muito código, colocar vários sistemas no ar, mas nenhum deles gerar impacto no negócio, ou mesmo nenhuma contribuição nossa gerar impacto na vida das outras pessoas desenvolvedoras do nosso time. Com o tempo, você começa a identificar dentro da organização quais são os projetos que realmente geram valor, mas antes de conquistar essa maestria, você sempre pode contar com o direcionamento da sua gestão ou de quem está em níveis acima de você.

É importante começar a perceber quando a nossa ambição pessoal está se sobrepondo às necessidades da organização. Trocar de tecnologia é muito legal, fazer algo novo com tecnologias novas é extremamente divertido, mas vale a pena fazer essa mudança no momento atual? Vale a pena reduzir a velocidade em que lançamos novidades nos produtos neste momento? Como está o mercado no qual sua organização está inserida? A concorrência está na frente ou atrás? Como está o momento financeiro da empresa? Temos dinheiro para investir na troca de tecnologia?

No final do dia, tudo que fazemos como pessoas desenvolvedoras de software é construir as estruturas para que o produto se sustente para o longo prazo. Se vamos trocar os trilhos com o trem passando em cima, precisamos saber exatamente o que e por que estamos fazendo.

8.3 Investindo o nosso tempo no que importa

Sabendo o que causa impacto na organização, começa o trabalho de estar no lugar certo e na hora certa. Talvez ninguém diga a você o

que fazer a partir de agora, e entender onde investir o seu tempo pode ser extremamente difícil.

A maior dificuldade de pessoas que não sabem onde devem colocar sua energia dentro de uma organização é a falta da **visão do todo**, o que algumas pessoas chamam de *big picture*.

Imagine que, no sistema em que você trabalha, você consegue saber que a parte crítica dessa aplicação é aquele módulo (ou módulos) do qual várias partes dependem, ou o qual é o ponto inicial de fluxo de dados para dentro da sua arquitetura. Mas você só sabe disso porque você realmente conhece esse sistema, conhece cada parte dele e sabe que este é um ponto crítico.

Entender como a organização está funcionando será similar, porém em um nível mais alto e mais abstrato. Você vai precisar conversar com pessoas acima de você e com outros times a fim de conectar todas as pontas para descobrir quais são os pontos mais críticos da organização.

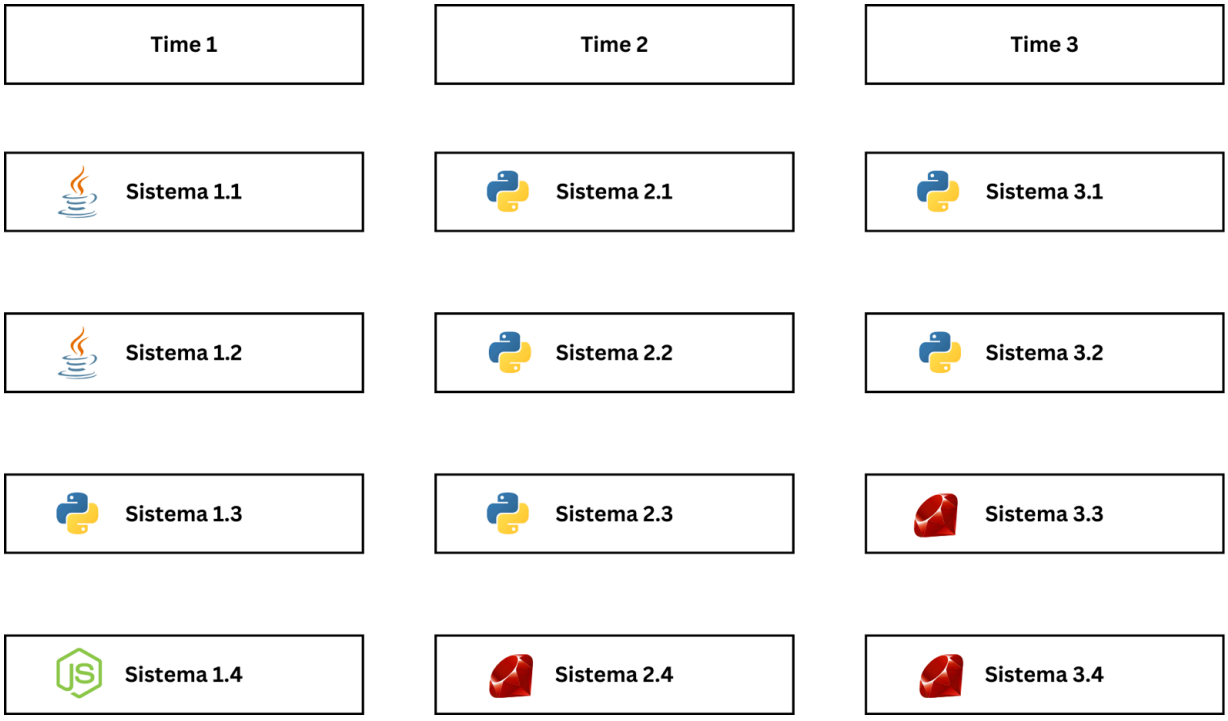
Pense a empresa como camadas: a primeira camada é onde você está como sênior, temos alguns sistemas, alguns times conectados, alguns times dependendo um do outro ou nenhuma dependência. Na segunda camada, temos grupos de times inteiros que precisam se comunicar de algum modo e sistemas que conversam entre si e dependem do funcionamento para continuar existindo. Se subirmos mais ainda de camada, veremos que esses grupos agora são maiores e talvez a falha de uma das partes pode gerar problema no processamento dos dados e quebrar todo um fluxo de algum produto ou até mesmo de vários produtos.

Para conhecer tudo isso e como as peças se conectam, será necessário ter muita conversa e muita conexão internamente. Podemos começar buscando reuniões com as pessoas que estão há mais tempo na empresa ou com quem está acima de nós na hierarquia e conhece como tudo funciona. Só assim conseguimos realmente entender o que é importante para a organização, pois

muitas vezes temos uma visão distorcida do que é importante e em conflito com o que nós acreditamos — e que nem sempre é a realidade.

Pode ser que alguém já tenha passado por esse caminho antes de você e já tenha criado um organograma que lhe pode ser útil nesse momento (por isso, minha primeira dica é conversar com outras pessoas mais experientes e com mais tempo de casa), mas pode ser que isso não exista. Uma sugestão pessoal é que você faça esse desenho para facilitar seu raciocínio. Pode ser algo muito simples, como um diagrama conectando caixinhas e explicando como os sistemas estão se comunicando, como está organizada sua infraestrutura, quais são as tecnologias que utilizam etc.

No exemplo a seguir, eu trouxe um caso que um amigo compartilhou comigo. A pessoa em cargo de *principal* identificou alguns pontos na engenharia que poderia trazer sérias dores de cabeça para a companhia em um futuro próximo. O que foi identificado por essa pessoa é que existiam muitos microsserviços dentro do ecossistema utilizando tecnologias diferentes, e eu vou explicar mais adiante o que ela achava ruim neste cenário.



A imagem é meramente ilustrativa, mas o que esse amigo comentou foi que existiam diversos serviços em Java, outros em Python, alguns times começando a utilizar Node.js, outros experimentando Ruby e alguns outros testando ainda outras tecnologias, como Golang e Elixir. Isso pode não soar negativo para nós, pessoas engenheiras, afinal de contas adoramos experimentar novas tecnologias e reescrever coisas para que elas fiquem melhores. Porém, para uma empresa, isso pode significar perda do que o mercado chama de *time to market*, o momento certo de lançar um produto ou funcionalidade que pode garantir maior penetração de mercado para a organização.

A explicação pela qual isso pode acontecer é o fato de que, se não tivermos algumas tecnologias homologadas para uso na empresa, podemos precisar, a todo momento, reescrever trechos de códigos que poderiam ser compartilhados entre times, módulos para aplicações no servidor ou componentes de interface que poderiam simplesmente ser instalados. Além disso, podemos precisar garantir segurança em algum pedaço dos sistemas e, ainda, recriar *pipelines* para a infraestrutura a cada nova tecnologia lançada. Isso é muito tempo desperdiçado, caso não exista um excelente motivo para acontecer.

Se a empresa trabalha com integração de terceiros em sua plataforma através da escrita de SDKs, por exemplo, em algum momento existirão diversos projetos com centenas de linguagens diferentes. Mas se é um produto digital, como uma aplicação web, um aplicativo ou uma plataforma, faz sentido que tenhamos uma certa uniformidade nas ferramentas que usamos para garantir um bom ferramental para todos.

Algumas empresas possuem até mesmo uma lista do que é utilizado para a construção dos sistemas, e isso só muda depois de passar por muita experimentação e construção de diversos facilitadores para o dia a dia de quem vai trabalhar na ponta final, que é a construção do produto em si.

> Nem sempre escrever código é a melhor solução para os problemas que a organização enfrenta. Apaixone-se pelos problemas que resolvemos, não pelo código que você escreve.

Uma dica sobre tecnologias: em mais de um lugar em que trabalhei, nós utilizávamos a prática de criar um ***technology radar*** para tentar garantir um equilíbrio entre as tecnologias que estávamos utilizando na empresa. O *tech radar* é um modelo criado pela ThoughtWorks de documentação das tecnologias utilizadas, observadas ou que devem ser evitadas na organização. Normalmente temos este documento disponível em algum lugar interno na empresa, mas ele também pode ser uma página em nosso site para mostrar para o mundo o que utilizamos e compartilhar conhecimento.

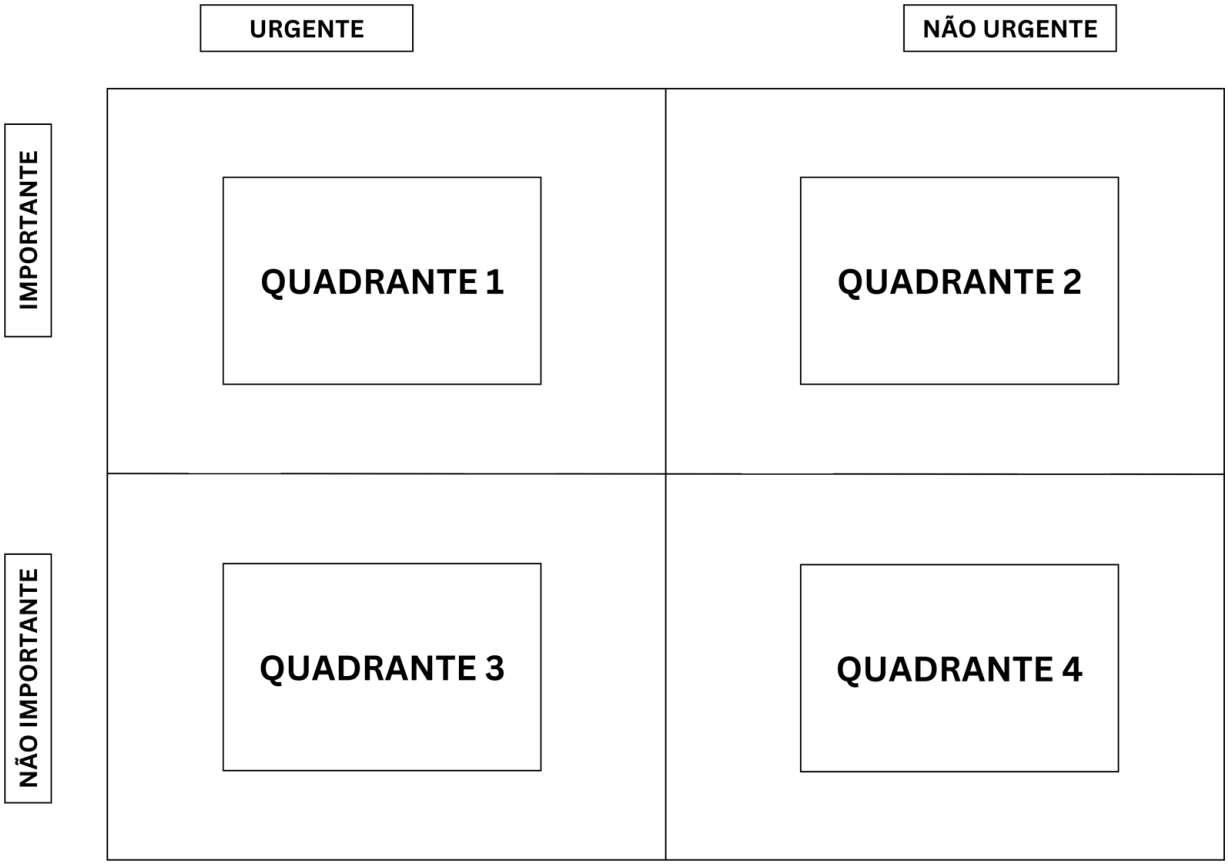
O *tech radar* mostra para a organização como um todo quais são as ferramentas em que estamos investindo energia no momento, quais nós já testamos, criamos templates, integrações com a infra e arquiteturas internas, e já podem ser utilizadas por todos os times. Caso você queira conhecer o *tech radar* da ThoughtWorks, acesse: thoughtworks.com/en-br/radar.

Este é só um dos exemplos de que, se você conhecer a empresa como um todo, pode começar a identificar pontos de atenção. Pode ser algo que precisa ser ajustado, algo que precisa ser mudado ou que precisa de uma monitoria mais intensa, como o uso de ferramentas de monitoramento com mais logs, por exemplo.

A partir do momento em que nos tornamos especialistas, escrever um código pode não ser a tarefa mais impactante que vamos fazer. Identificar esses riscos é mais fundamental para o nosso dia a dia do que escrever o código de uma nova funcionalidade no produto final.

Uma técnica que utilizo para encontrar coisas realmente importantes para fazer é seguir a Matriz de Eisenhower. Essa matriz é basicamente uma forma de organizar as tarefas por ordem de

urgência e importância. Afinal, algo urgente nem sempre é importante e algo importante nem sempre é urgente. A seguir, temos uma imagem de como essa matriz funciona:



- No quadrante 1, colocamos as coisas que realmente devemos fazer, pois elas são importantes e urgentes;
- No quadrante 2, colocamos as coisas que devemos agendar para fazer depois, pois elas são importantes (então não podemos deixar de fazer), mas não são urgentes;
- No quadrante 3, colocamos as tarefas que podemos delegar para alguém (ou para alguma automação, por exemplo), pois elas não são importantes, mas são urgentes;
- No quadrante 4, temos as tarefas que não devemos fazer e/ou podemos jogar fora, pois não são importantes nem urgentes.

Você pode fazer um teste agora mesmo pensando em tarefas pessoais, organização da casa ou qualquer coisa que pareça estar sobrecarregando você. Esse é um exercício crítico para identificar o

que é realmente urgente, e que, se não for feito, pode causar um grande problema, e o que é importante, que realmente precisa ser feito e não pode deixar de acontecer por nada.

Quando identificar que determinado ponto é crítico para a organização, você precisa se colocar nas discussões sobre este assunto. Precisa se conectar com os times e projetos que podem sofrer alguma consequência ou mesmo gerar um problema.

Em determinado momento da minha carreira, estava acontecendo a construção de um sistema integrador que seria responsável por tudo relacionado ao usuário dos produtos de uma empresa onde trabalhei. Está muito claro o quanto isso é impactante para a organização. Este sistema mudaria a forma como todos os sistemas se conectavam, assim como poderia causar a indisponibilidade para vários produtos, caso fosse implementado sem uma boa comunicação com todos os times da companhia. Naquele momento, eu atuava como tech lead. Mais da metade da minha semana era participando da construção desse sistema, direta ou indiretamente, fosse com sugestões ou com meus medos quanto à utilização do sistema pelo meu time e aos produtos pelos quais eu era responsável.

Em outros momentos, as iniciativas já estavam acontecendo e ninguém me chamou para participar daquilo. Eu identifiquei que elas poderiam atingir minha tribo e comecei a participar ativamente para ajudar a garantir que a minha unidade de negócios não fosse impactada negativamente. Talvez ninguém chame você para participar de algo porque, realmente, são muitos times, e existem times que geram mais impacto para a organização do que outros. Se o seu não é tão importante, pode ser que você fique de fora, mas você precisa se colocar nesses espaços. Não é hora de sentir raiva ou ficar com o ego ferido porque as pessoas não acham o seu time tão essencial. Talvez não seja mesmo. Existem poucos times dentro das organizações que realmente estão gerando muito lucro e precisam de muita atenção. Essa classificação não é o que nos

interessa, o que nos importa é estar no lugar certo para garantir a geração de valor para o nosso grupo e para a organização.

Apesar de tudo isso, em alguns momentos precisamos contar com a nossa intuição, curiosidade e até mesmo os problemas pessoais que gostaríamos de resolver. Em alguns momentos essa intuição será responsável por catalisar o nosso foco em algo que ninguém está dando atenção, mas pode causar um impacto imenso.

José Valim, criador do Elixir, comentou comigo que a linguagem foi criada devido a uma **curiosidade pessoal**, e não um projeto da empresa onde ele trabalhava para construção de algo que se tornaria tão impactante ou mesmo alguém falando para ele que deveria fazer isso. Enquanto ele era *core commiter* do Ruby e do Ruby on Rails, parte dos bugs mais complexos de resolver e reproduzir eram relacionados à concorrência. Foi justamente na época em que concorrência começou a ser um assunto muito comentado e o tema chamou sua atenção; logo, ele quis resolver estas questões da comunidade Ruby, e foi com a criação dessa linguagem que ele conseguiu solucionar parte dos problemas e ainda transformou a indústria de software no mundo inteiro.

O mesmo acontece com Thiago de Arruda, criador do Neovim, um editor de textos também utilizado no mundo inteiro por diversas pessoas desenvolvedoras. Thiago já era usuário do editor Vim há alguns anos e ele percebeu que a maior reclamação dos usuários do editor era entre criadores de plug-ins para a ferramenta. Eles comentavam que era muito difícil fazer a comunicação assíncrona com o Vim, e ele decidiu tentar resolver este problema. Parte da solução gerou a criação do novo editor, o Neovim, que se tornaria o sucesso que é hoje em dia.

Há momentos em que uma necessidade compartilhada entre setores pode fazer com que a nossa indústria se mobilize na construção de alguma tecnologia que resolva determinado problema, como foi no caso da linguagem Lua, criada pelo professor Roberto Ierusalimschy e que, segundo ele, foi feita para ajudar a configurar softwares

utilizados na indústria petrolífera. Lua se tornou uma ferramenta mundialmente utilizada em vários setores, como a indústria de jogos.

Ou seja, em alguns momentos, o trabalho é extremamente prático; em outros, será intuitivo. Precisamos estar muito bem preparados no sentido de conhecimento teórico e prático do que fazemos para que, quando uma necessidade surgir, tenhamos as ferramentas certas para atuar nela.

O professor Roberto é muito enfático quando diz que precisamos estudar muito para que estejamos preparados no momento em que os problemas aparecem. O José Valim e o Thiago são dois exemplos de pessoas que estavam extremamente preparadas e resolveram um problema real e latente porque sabiam como fazer. O mais legal é que nenhuma dessas pessoas imaginava que o impacto do seu trabalho seria tão grande: eles só queriam resolver um problema.

8.4 Trabalhando de maneira eficiente

Infelizmente, o nosso tempo é limitado e não conseguimos fazer tudo o que desejamos, e isso também se aplica para o dia a dia como especialistas. Gostaríamos de estar junto com todos os times que precisam de nós, resolver todos aqueles bugs que nós já sabemos como resolver rapidamente, contribuir para uma boa experiência de trabalho em todas as pontas, mas isso é impossível.

Atuar como especialista fica muito difícil se não possuirmos certo nível de organização pessoal. Se entramos em reuniões que não são importantes, atrasamos o trabalho no qual realmente faríamos a diferença. Quando passamos muito tempo em uma atividade que poderia ser feita por outra pessoa, acontece o mesmo problema de investimento de energia onde não deveríamos estar.

Existem algumas técnicas que eu utilizo para trabalhar de maneira mais organizada e pensar melhor onde investir meu tempo. Vou compartilhá-las aqui, mas estas são técnicas que funcionam para uma parcela de pessoas, nem sempre podem funcionar para você. Será necessário que, além de estudar tecnologias, agora você também estude sobre produtividade, organização pessoal e até mesmo sobre gestão de pessoas para conseguir delegar trabalho para outras pessoas. Pessoas especialistas precisam ser muito mais pragmáticas do que as demais, senão o prejuízo para a organização é muito grande.

Algo que eu sempre recomendo para todas as pessoas, e em qualquer nível de sua carreira, é buscar um alinhamento muito claro com a sua liderança sobre qual é a sua atuação e seu escopo de trabalho. Até onde você pode ir e onde não deveria investir sua energia. Com uma rápida conversa com a liderança, conseguimos eliminar os problemas de concepção que podemos ter sobre qual é o nosso trabalho e o que a empresa realmente espera de nós. Fazer isso vai nos ajudar até mesmo a conseguir saber quando negar participação em um projeto ou conseguir escapar de reuniões improdutivas.

Caso você nunca tenha pensado nisso, pode utilizar este livro como referência na hora de conversar com sua gestão. Você pode utilizar o conhecimento que adquiriu sobre qual é o papel da pessoa especialista para pensar em conjunto o que você deve fazer no seu dia a dia.

Existe uma certa liberdade no cargo de especialista que traz um peso grande de responsabilidade, que é o fato de nosso escopo nem sempre poder ser fechado em tarefas específicas. Na maioria das vezes nós temos liberdade de procurar onde tem problema e investir nosso tempo nisso. Mas, antes de sair atacando essas partes, converse com sua liderança e com outras pessoas se seu palpite está correto ou até mesmo se a organização realmente pode se beneficiar disso no futuro. Nem sempre nossa intuição estará correta.

Justamente pelo fato de que nem sempre vamos acertar, é muito importante evitar o *achismo*, pois nossas decisões impactam muitas vidas. Uma decisão errada de pessoas especialistas pode aumentar muito o custo de uma empresa ou derrubar um sistema por tempo suficiente para causar um prejuízo financeiro tão grande que vai tirar o emprego de pessoas de várias partes de uma organização.

Sempre se comunique, busque outras opiniões e dados sobre as hipóteses que está levantando. Apoie-se na literatura e no que a indústria tem feito com bons resultados. As pessoas adoram como eu sempre tenho um artigo ou um *blog post* (ou vários) sobre os assuntos dos quais eu converso e quando estou tentando defender um ponto. Isso é essencial para garantir que estamos no caminho certo.

É importante nos apoiarmos no conhecimento que a indústria já possui, porém também devemos investir nosso tempo em experimentação, quando o que a indústria fala não nos convence. Busque sempre ter uma boa quantidade de informação sobre algo antes de qualquer movimento, e essa quantidade só o seu contexto vai dizer, pois depende de quanto tempo você tem disponível até a tomada de decisão, assim como a quantidade de recursos para apoiar você neste momento.

Mas somente o que comentei até agora não vai nos garantir eficiência no dia a dia de trabalho. Lembra que falamos sobre organização pessoal? Sabe aquela reunião que você não tem certeza se deveria fazer parte? É hora de aprender a perguntar antes de aceitar um convite. Sempre que chegar aquele convite por e-mail, caso você tenha dúvidas de sua importância dentro daquele evento, pergunte se sua presença se faz realmente importante nessa discussão. Na maioria das vezes vão dizer que sim, e você precisará questionar caso esse "sim" não seja suficiente para lhe convencer que você realmente deva estar lá. Não tem problema nenhum em fazer isso. As pessoas podem se incomodar, mas elas vão entender se você utilizar a empatia para se comunicar e dizer que precisa selecionar melhor as agendas, pois está com muitos compromissos

em paralelo (que é o estado normal da agenda da pessoa especialista). Talvez seja até mesmo o caso de indicar outra pessoa para participar no seu lugar.

Por muitas vezes eu sou convidado para reuniões pelo fato de ser uma pessoa mais sênior dentro dos times. Mas que diferença eu vou fazer, por exemplo, em uma reunião de resolução do problema de alta performance de um algoritmo se eu não sou a pessoa que sabe algo sobre isso? Eu sou péssimo nisso. Pode acontecer de eu receber um convite para discussões de arquitetura, mas eu ter um problema urgente de segurança para atender; nesse caso, eu posso indicar alguém para estar no meu lugar para ser a facilitadora nesta reunião.

Mas essa abordagem nem sempre resolve nosso problema. Para evitar que essas reuniões impactem negativamente meu tempo, eu crio compromissos na minha agenda nos quais eu vou investir no que eu realmente sei que vai gerar valor para a organização. Pode ser uma revisão de um código crítico, pode ser a conversa com times que estão desenvolvendo algo que eu preciso acompanhar ou mesmo o compartilhamento de conhecimento sobre alguma prática que eu gostaria que os times começassem a utilizar.

Dependendo de qual é sua atuação onde trabalha agora, talvez seja bom deixar algumas reuniões de alinhamento recorrentes com alguns times, além de um ciclo de feedback para direcionar as pessoas para o caminho certo ou comentar quando estão cometendo algum erro. Isso também garante que você tenha tempo para conversar com as pessoas que precisam de você em vez de cair em conversas eternas que poderiam ser feitas com outros profissionais dentro da organização.

Além disso, eu costumo deixar espaço para escrever código, afinal de contas, nós temos o contexto da empresa como um todo, nós realmente sabemos quando a criação de um módulo ou a modificação de algo vai gerar um grande impacto para a organização, mas, com tanta reunião, nos falta tempo para escrever

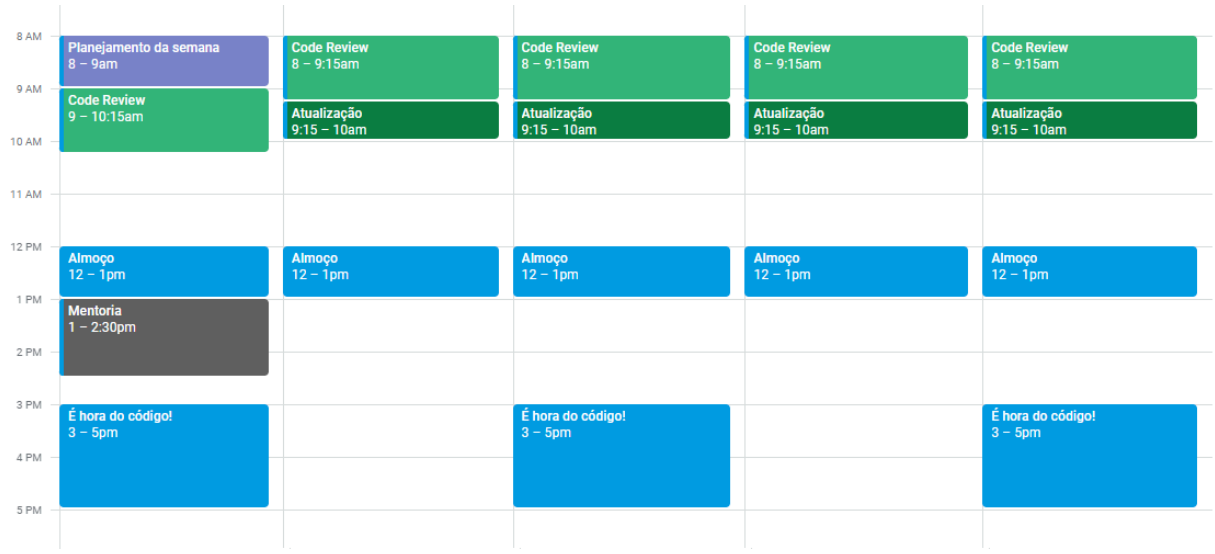
este código ou mesmo conceber uma documentação para que alguém o faça.

Tenha em mente que sua agenda precisa ser organizada de verdade a partir de agora, que ler os e-mails da empresa, entender os relatórios de produto e manter uma boa comunicação é o que vai garantir eficiência ao invés de gerar retrabalho e sobrecarga.

Dependendo do que você está fazendo atualmente, talvez tenha que participar de entrevistas de emprego. Tenha também um slot reservado para isso em sua agenda para que o time de recrutamento utilize este espaço.

Ainda sobre esse assunto de agenda, organize um tempo para compartilhar seu conhecimento com alguém. Eu deixo livre uma hora por semana para que alguém entre em contato comigo para conversarmos sobre carreira ou assuntos técnicos que as pessoas não entendem e sabem que eu consigo ajudar. Isso é muito eficiente para criar um ambiente mais equalizado em nível de conhecimento, além de criar uma rede de pessoas que realmente sabem o quanto você domina algo e que podem confiar no que você diz quando traz uma hipótese ou toma uma decisão.

Veja um exemplo de como fica a minha agenda com meus compromissos fixos:



- Toda segunda-feira, a primeira coisa que eu faço é um planejamento da semana, qual será meu foco/objetivo central;
- Todos os dias, possuo um momento de *code review*, no qual reviso os *pull requests* de projetos em que estou conectado;
- Todos os dias de manhã, tento me atualizar com coisas que estão acontecendo na empresa, seja por e-mails, chats como o Slack, conversa com a gestão ou qualquer outro modo de comunicação;
- Meu horário de almoço está fixado, para que ninguém me tire a paz neste momento de felicidade do trabalhador;
- Toda segunda-feira deixo um slot vazio para que qualquer pessoa possa agendar uma conversa comigo depois do almoço para conversarmos sobre carreira;
- A cada dois dias, tenho um momento no qual posso focar em escrever código até o final do dia.

É claro que essa agenda muda de acordo com a prioridade do momento. Mas eu procuro seguir isso à risca, evitando mudar meus compromissos em troca de reuniões onde não precisam de mim.

Quando você atua como tech lead, sua agenda terá de possuir mais slots para conversar com seu time, além de que, quando estamos em algum projeto, a maior parte da nossa agenda é voltada para aquele projeto, muitas vezes tirando até mesmo o tempo livre no qual poderíamos escrever código.

Mas, apesar da agenda "travada", as reuniões sempre vão acontecer e precisam ser organizadas, para que você não perca seu tempo e para que as outras pessoas não percam o tempo delas. Algo que eu utilizo desde que fiz um curso de gestão de empresas (sim, mesmo como desenvolvedor, eu invisto tempo em vários tópicos diferentes porque eu gosto, mas você não precisa fazer isso) foi começar a ter uma pauta, começar a reunião sempre comentando sobre essa pauta e tentar manter o foco nisso.

Com uma pauta bem definida, sabemos como podemos direcionar a conversa. Se vamos falar sobre como negligenciar a segurança da

informação agora pode nos gerar prejuízos imensos no futuro, conseguimos trazer informações que nos ajudam nesta argumentação, assim como direcionar o que, dentro do gigante mundo de segurança, vamos tratar. Isso não significa que, caso alguém fale algo fora desse assunto (por exemplo, performance), nós vamos cortar essa pessoa e não dar espaço para ela demonstrar o quanto ela também acha importante falar sobre o tema. Nós podemos combinar com essa pessoa para que ela puxe uma outra reunião para discutir sobre performance em outro momento e, por agora, continuamos falando sobre segurança da informação. Isso não significa ser rude com essa pessoa, é assumir a responsabilidade por garantir a qualidade da discussão.

Uma vez eu vi alguém fazendo algo muito legal depois de uma reunião. Para contextualizar: durante essa reunião, teve uma pessoa que trazia muitas dúvidas relacionadas à tecnologia que estávamos discutindo, mas nenhuma sobre o problema que gostaríamos de resolver. Eu, no auge da minha imaturidade, já estava ficando incomodado com aquela pessoa, mas o que a liderança fez foi essencial a meu ver. No final da reunião, nós recebemos um e-mail com as perguntas que aquela pessoa fez e as respostas extremamente bem estruturadas. Essa é uma prática muito construtiva para times de engenharia, que, muitas vezes, "viajam" em vários assuntos não relacionados à pauta de uma reunião, mas que são pontos importantes para aquela pessoa que está questionando.

Algo bem interessante sobre esse comportamento é que isso funciona muito bem pra mim também. Eu tenho ansiedade generalizada e nem sempre consigo focar em tudo o que está sendo dito na reunião. Em determinados momentos, eu faço perguntas que não fazem sentido e fico sem resposta. Alguém anotar minha pergunta e responder depois, além de extremamente útil, torna o ambiente de trabalho muito mais acessível.

A última dica sobre reuniões é um pouco mais complicada, pois determinar uma pauta e anotar perguntas para responder depois é

algo mais "interessante" para quem lidera a sessão. Mas existe uma prática muito eficiente para garantir o alinhamento geral, que é enviar um e-mail com o resumo da reunião ou documentar esse resumo em algum lugar para que todas as pessoas possam confirmar se realmente entenderam o que foi decidido.

Em discussões de engenharia, conseguimos utilizar práticas como as anotações arquitetura ou documentação de software para guardar o que decidimos, mas em reuniões gerais isso é um pouco mais complicado. Pode dar mais trabalho, mas eu garanto que isso evita muito retrabalho no futuro.

Com essas dicas, nós já conseguimos manter o trabalho mais focado, porém será necessário também criar suas próprias maneiras de conseguir essa eficiência com base no seu contexto profissional e organizacional.

8.5 Criando espaço para o crescimento de outras pessoas

É muito comum que pessoas técnicas se tornem *tecnosolucionistas*, pois, como estamos buscando sempre o máximo de eficiência em tudo o que fazemos, começamos a acreditar que qualquer coisa se resolve com algoritmos. Nós paramos de enxergar os problemas reais como coisas que, às vezes, não se resolvem com linhas de código. Isso impacta não somente o nosso crescimento profissional, mas também a organização para a qual estamos trabalhando.

Tecnosolucionismo, ou solucionismo tecnológico, é uma ideologia identificada por Evgeny Morozov nos discursos que provêm do Vale do Silício. Essa ideologia tenta fazer as pessoas acreditarem que é possível solucionar **qualquer problema** com tecnologia. Porém, alguns problemas do ser humano só são resolvidos através de ações humanas.

Existirão momentos em sua carreira que enviar uma mensagem no Slack não vai finalizar uma crise, vai começar outra. Em outros momentos, você vai acreditar que enviar um e-mail é mais útil do que fazer aquela reunião, porém existem pessoas de todos os tipos trabalhando com você, aquelas que aprendem e também conseguem performar através de abstração e leitura e outras que precisam de conversa e participação ativa na construção de um raciocínio pessoalmente.

Se nos fecharmos para as relações interpessoais, estaremos fadados a criar um ambiente que não é inclusivo, é ineficiente e não é escalável para o longo prazo. Nenhuma empresa grande fica no ar sem uma boa comunicação interna ou boas relações humanas lá dentro.

Esse comportamento de transformar tudo ao redor em zeros e uns nos tornam pessoas rígidas, que não aceitam quando as outras erram e não estão lá quando os outros precisam. E isso, ao longo prazo, vai criar um ambiente medíocre de trabalho, com alta rotatividade e pessoas fazendo o mínimo só para garantir seu emprego em vez de estarem motivadas a trabalhar bem para também se manterem em crescimento.

Nós, como especialistas, somos um exemplo que outras pessoas querem olhar e seguir. É importante se mostrar à disposição para tirar uma dúvida, ouvir uma reclamação e compartilhar conhecimento. Quando as pessoas ao redor estão evoluindo, seja tecnicamente ou pessoalmente, tudo ao redor também evolui junto. Isso causa impacto direto em você hoje, gerando bons resultados onde você aparece, e, futuramente, na organização, com profissionais altamente qualificados criando valor. Para individualistas, isso significa promoção e mais dinheiro na conta. Para coletivistas, isso significa crescimento do coletivo como um todo.

É importante que sejamos pessoas positivas, e ser alguém positivo não significa ser iludido e desconectado da realidade. É sobre o fato

de você não agir como uma âncora: só enxergando os pontos ruins em tudo o que acontece ao redor e levando todo mundo para baixo junto com você. Precisamos perceber quando, em vez de agradecer alguém pelo trabalho que fez, só comentamos os pontos onde a pessoa tem que melhorar. Todo mundo precisa melhorar, mas todo mundo também precisa de uma massagem no ego para aumentar a autoestima. Sem autoestima, não há motivação para realizar as tarefas chatas do dia a dia no mercado de trabalho.

Só que ser positiva não é o mesmo que dizer sim para tudo. Sem bons fundamentos, uma ideia não deve ir adiante (a não ser que seja algo em que você, seu time ou a empresa queira apostar). Nós conseguimos conviver com o fato de que nem tudo o que propomos irá para frente e que nem tudo o que desejamos se realizará. Porém, no dia a dia de trabalho, é importante não eliminar a motivação das pessoas. O crescimento só vai acontecer se nós conseguirmos mostrar para as pessoas que existem coisas legais no futuro e que o que elas querem agora talvez não seja realmente viável — talvez nunca seja, mas isso se faz através da comunicação.

8.6 Como é a entrevista de emprego para pessoas especialistas

Responder como é o processo de entrevista para pessoas especialistas soa muito similar a quando me perguntam sobre esse momento em qualquer posição: não existe uma resposta certa. Cada empresa busca alguma característica, por isso nem sempre conseguimos nos preparar para elas. Existem coisas similares entre várias organizações, que é o que eu compartilho neste livro. Porém, é importante que você saiba que isso aqui não é um checklist para que você passe em entrevistas, é um relato de como as coisas funcionam e como você pode buscar se preparar para este desafio, mas sem criar falsas expectativas.

O processo de entrevista como um todo deve possuir: o contato com a pessoa de recrutamento, seguido de uma conversa com a gestão que está buscando alguém; desafios de código; uma conversa ou desafio de arquitetura de software; a conversa com outras pessoas especialistas; e, por fim, a conversa de *fit* cultural e a oferta em si.

Durante a conversa com a pessoa de recrutamento, será passada uma verdadeira lista de coisas que a gestão está buscando. Não existe nenhum padrão aqui, pois algumas vezes respondemos perguntas muito simples como "*quanto tempo de experiência você tem com x, y, z*"; em outras, a pessoa quer que listemos projetos de alto impacto em que trabalhamos e qual era a nossa função nesses projetos. Em todos os casos, prepare-se para comentar quando, como e por quanto tempo você usou alguma tecnologia, de que forma a utilizou e explicar profundamente sobre como ela funciona, além de guardar em seu histórico de vida como você trabalhou nos projetos de alto impacto dos quais fez parte.

Uma técnica para lembrar o que eu fiz nos meus projetos importantes é a escrita de um *brag document*, que aprendi através de um post de Julia Evans (https://jvns.ca/blog/brag-documents/). O *brag document* é, basicamente, um histórico de coisas legais que gostaríamos de lembrar no futuro. No artigo original, ela comenta ainda sobre o fato de que nossa gestão também não vai lembrar de coisas importantes que fizemos, então fazer isso nos ajuda até mesmo para uma futura promoção.

Eu costumo fazer algo bem simples: adiciono a data de quando algo aconteceu em formato de mês e ano, uma breve descrição do que era para ser feito, uma descrição do que eu fiz e um outro texto sobre quais foram os resultados do meu trabalho. Caso seja possível, coloco links para compartilhar depois, como tarefas do Jira, *issues* do GitHub, *pull requests* ou RFC aprovada etc. Por exemplo:

Data	Desafio	Realização	Resultados
Julho/2022	Compartilhar componentes de interface como aplicações autocontidas (com a lógica de negócio) entre os times da unidade de negócio	Minha proposta inicial foi a utilização de microfront-ends, mas a empresa não gostaria de investir nisso para o longo prazo, portanto cheguei à solução de utilizar *monorepos* para criar componentes autocontidos exportáveis entre as aplicações criadas dentro deste mesmo repositório.	A solução foi aceita entre outros times de engenharia e gestão e, depois de implementada, possibilitou a criação de três aplicações antes do prazo determinado pelo time de produtos.

Em uma entrevista, você pode comentar exatamente o que está escrito. Em um ciclo de avaliação, você pode utilizar esse texto para lembrar sua gestão do impacto do seu trabalho.

Continuando o processo de avaliação, o próximo passo será a entrevista com a gestão. As perguntas serão similares (ou as mesmas) feitas pela pessoa do recrutamento, porém agora com mais tempo para conversar e com mais dúvidas para aprofundar sobre o que nós falamos. Então, se levarmos em consideração o resultado do meu brag document acima, as perguntas da gestão

seriam coisas como: por que você queria utilizar microfront-ends, como você se sentiu quando as pessoas não quiseram seguir sua primeira ideia, como você chegou na segunda solução, como você escolheu e qual foi a tecnologia utilizada para construir os monorepos etc.

A pessoa de gestão também está interessada em como você resolve problemas, não somente em como você escreve código; por isso, prepare-se para compartilhar a forma como você realmente pensa.

O próximo passo são os testes de código a que já estamos acostumados, porém com um peso maior na qualidade com a qual você os escreve. Não será cobrado que você escreva código simples com padrões de projetos, algoritmos clássicos ou qualquer coisa muito teórica, mas se espera que você saiba justificar o que está fazendo e com bons argumentos. Se não vai utilizar o melhor algoritmo já conhecido para resolver aquele problema, por que você chegou a essa conclusão? Se vai utilizar este algoritmo em vez de escrever uma solução mais simples para esta primeira entrega, por que você fez isso? Essas respostas são importantes para entender como você trabalha no dia a dia e como você vai impactar a empresa, não para saber o quanto de teoria de ciência da computação você sabe. Você pode passar por um, dois ou muitos desafios de código em um mesmo processo seletivo ou até mesmo na entrevista. Recomendo fortemente que se prepare para isso.

Uma boa maneira de se preparar para os desafios que você pode encontrar em entrevistas e processos seletivos é resolver problemas de código em plataformas de programação competitiva, como o HackerRank e LeetCode. Como dica de leitura, recomendo o livro *Cracking the Code Interview*, de Gayle Laakman McDowell (2015), que traz várias dicas para mandar bem nessa etapa, além de uma série de desafios de código utilizados em processos seletivos reais que a autora catalogou. Esse livro é recomendado até mesmo por grandes empresas em seus processos seletivos.

A entrevista sobre arquitetura (ou *system design*) também não é diferente daquilo que você já faz como sênior, mas, de novo, há muita ênfase nos porquês das suas tomadas de decisão e sempre pensando no impacto do que você faz. Para se preparar para estas, eu recomendo os livros da série *System Design Interview*, de Alex Xu, publicados de maneira independente.

Os próximos passos — a conversa com outras pessoas especialistas, a conversa de *fit* cultural e o papo sobre a oferta em si — acontecem para que as pessoas que vão trabalhar conosco nos conheçam e tenham certeza de que desejam nos ter por perto, portanto não tem nenhuma dica especial aqui. Você deve se apresentar, comentar sobre os assuntos que mais domina e mais gosta, além de compartilhar mais sobre sua experiência de trabalho com essas pessoas. Podem surgir perguntas pessoais nesta etapa também; caso não se sinta confortável para respondê-las, não tem problema dizer que prefere não as comentar.

As pessoas querem entender como você pode se encaixar no time para o qual está concorrendo ou como você pode agregar para a cultura da empresa. Conhecer um pouco mais sobre a organização pode lhe ajudar a comentar os pontos que são comuns entre você e quem já está por lá.

Algo extremamente importante e que pode ser crucial para a nossa felicidade no trabalho é: procure descobrir qual é o salário das pessoas especialistas na empresa para a qual está concorrendo. Caso não exista isso em nenhum lugar, como nos sites de carreira das empresas ou no Glassdoor, utilize a entrevista com recrutamento para validar isso. Na hora em que receber a pergunta "*qual sua expectativa salarial?*", você pode perguntar "*qual a faixa salarial para este cargo na sua empresa?*". Isso nos ajuda a não entrar com uma expectativa muito alta — e a contratação ficar impraticável — e, também, a não colocarmos um salário muito baixo.

8.7 Como se atualizar estando longe de código

Algo extremamente comum de acontecer com pessoas especialistas é ficar longe da prática de escrita de código por muito tempo. Isso nos desatualiza com possíveis novas convenções que a comunidade possa estar seguindo ou até mesmo com atualizações das linguagens. Quem trabalha com JavaScript, por exemplo, sabe que teve um boom de mudanças na linguagem com o lançamento do ES6, em seguida vieram vários ciclos de transformação da tecnologia por conta do uso no back-end com Node.js, depois surgiu o TypeScript e, de repente, milhares de empresas e produtos estão usando isso. É impossível acompanhar tanta coisa atuando em projetos tão complexos que sugam nossa energia e todo o nosso tempo disponível dentro do trabalho.

Pessoalmente, eu adoro a prática da leitura para me manter atualizado, por isso sigo vários blogs e pessoas que compartilham conhecimento, mas também existem os eventos e fóruns para quem não gosta da leitura de *blog posts*. Conteúdo é o que não falta para pessoas desenvolvedoras, mas o mais importante é saber seguir pessoas que realmente compartilham coisas importantes nas redes sociais. Muitas vezes vamos ouvir falar de algo que será utilizado daqui a alguns anos antes de se tornar popular. Ir a eventos também é uma ótima maneira de aprender, e não sou só eu que penso assim.

Eu ouvi este conselho de diferentes pessoas que são referência na nossa área, como **Otavio Santana**, *distinguished* na Zup e membro do comitê executivo da linguagem Java; **Hisham Muhammad**, criador do *htop*, uma ferramenta extremamente utilizada pelas pessoas desenvolvedoras e contribuidor na linguagem Lua; e o **Giovanni Bassi**, CSA (*chief software architect*) na Lambda3. Para essas três referências no mercado de tecnologia e com backgrounds tão diferentes, acompanhar pessoas que compartilham conteúdo importante, participar da comunidade de software e ir a bons

eventos faz muita diferença para prever os próximos passos na indústria.

O Giovanni Bassi, por exemplo, utiliza a mesma prática que eu: temos uma conta no aplicativo Feedly, onde conseguimos seguir vários blogs ao mesmo tempo e receber atualização sempre que algo novo é publicado. Todos os dias, eu abro o Feedly e leio diversos *blog posts* para me manter atualizado.

Por isso, faça um levantamento de quais são as pessoas relevantes da área que você deveria acompanhar. Pode ser que sejam pessoas que escrevem livros, que geram conteúdo para o YouTube, que têm blogs legais ou que falam muito em redes sociais, isso vai depender de como você aprende e como isso se encaixa na sua rotina.

Não consigo citar aqui exemplos de comunidades ou eventos dos quais você deveria participar, pois isso vai depender muito do seu interesse pessoal. Eu, por exemplo, adoro *meetups*, aqueles eventos menores nos quais levamos palestras, rodas de discussão e outros formatos mais dinâmicos. Curto bastante o fato de que a pessoa que está puxando um tema está muito próxima do público, e isso abre espaço para mais perguntas e uma partipação mais ativa das pessoas da plateia. Mas é nas grandes conferências que você vai encontrar conteúdos técnicos mais avançados e tópicos que podem se tornar de grande interesse geral nos próximos anos. Procure não consumir somente conteúdo de uma região ou localidade: neste nível profissional, é extremamente importante ampliar a diversidade de referências para acompanhar o que outras pessoas no mundo estão fazendo para resolver seus problemas de engenharia de software.

O que você gosta de fazer: ler um livro, ler um artigo, assistir a um vídeo? Descubra isso e vá atrás das boas referências que geram estes conteúdos. Isso vai garantir a sua atualização por muito tempo.

Conclusão

Assumir como especialista vai mudar muito a forma como trabalhamos. Precisamos aprender a lidar com imprevistos, agenda, conflitos de horários, problemas em produção, priorização e várias tarefas pequenas ou grandes que nunca lidamos com tanta importância.

Não precisa se desesperar por agora, se você vai assumir essa posição ou acabou de receber esse cargo, o tempo também vai trazer experiências que irão ajudar você a criar um modo de trabalho que traga eficiência para o seu dia a dia. Eu compartilhei diversas atitudes que me ajudam no dia a dia e que foram ensinadas através do exemplo de outras pessoas ao meu redor. Você também vai aprender muito e alcançar este equilíbrio.

O mais importante deste capítulo, para mim, é que algumas vezes os desafios serão práticos, reais e latentes em nossas vidas e, em outras, teremos que contar com a nossa intuição. Mas, para todos os cenários, precisamos nos preparar ativamente para enfrentar os futuros problemas.

Capítulo 9
Conclusão

O caminho até se tornar especialista será de muita descoberta pessoal e busca pelo entendimento sobre como a indústria está se comportando no momento em que estivermos prontos(as) para dar este passo, além da procura por uma boa mentoria e também um lugar legal para fazer essa transição para um nível profissional com tanta responsabilidade.

Independente do momento em que estiver, agora você já tem todas as ferramentas possíveis para ou assumir o papel de especialista ou começar a se preparar. Caso você assuma esse papel nos próximos meses ou anos, espero que seja uma ótima experiência, que transforme sua vida para melhor e que seja motivo de muito orgulho.

Se sentir que este lugar não é para você, não tem problema algum dar um passo atrás e voltar a atuar somente como sênior ou trocar de trilha para gestão. O mais importante é que você seja feliz em seu trabalho, pois você passa muito tempo nele. O meu objetivo com este livro é munir você de informação útil para que você tenha uma carreira mais equilibrada — e, para isso, você precisa estar bem com seu lugar de atuação.

Caso você tenha gostado da leitura, compartilhe o que aprendeu aqui com as pessoas da sua empresa ou organização e também com a sua comunidade. Para que tenhamos um ambiente mais favorável para o nosso futuro profissional, precisamos conversar mais e espalhar a informação de como as coisas funcionam.

Espero ter ajudado você até aqui! Desejo-lhe muita sorte em sua caminhada!

Capítulo 10
Referências

BARNES, Wendy; WILKINS, Morgan. GitLab Job Grades. In: The Compensation Calculator. *GitLab*, c2023. Disponível em: https://about.gitlab.com/handbook/total-rewards/compensation/compensation-calculator/#gitlab-job-grades. Acesso em: 18 ago. 2023.

BETTER. *Career Ladder*. c2023. Disponível em: https://better.engineering/career-ladder/. Acesso em: 18 ago. 2023.

BOCCHINI, Bruno. Mulheres têm rendimento 21% inferior ao dos homens, mostra pesquisa. *Agência Brasil*, 2023. Disponível em: https://agenciabrasil.ebc.com.br/economia/noticia/2023-03/mulheres-tem-rendimento-21-inferior-ao-dos-homens-mostra-pesquisa. Acesso em: 18 ago. 2023.

BOEHM, Camila. Negros enfrentam mais dificuldades que brancos no mercado de trabalho, diz MPT. *Agência Brasil*, 2017. Disponível em: https://agenciabrasil.ebc.com.br/geral/noticia/2017-11/negros-enfrentam-mais-dificuldades-que-negros-

no-mercado-de-trabalho-diz-mpt. Acesso em: 18 ago. 2023.

BRITO, Daniel. Trilhas de carreira não lineares: o desenvolvimento profissional. *ESPM*, 2021. Disponível em: https://blog.espm.br/pos/trilhas-de-carreira-nao-lineares-o-desenvolvimento-profissional. Acesso em: 18 ago. 2023.

BRITO, Sabrina. Estudo: Heterossexuais ainda ganham mais do que LGBTs. *Veja*, 2021. Disponível em: https://veja.abril.com.br/ciencia/estudo-heterossexuais-ainda-ganham-mais-do-que-lgbts. Acesso em: 18 ago. 2023.

CARMO, Jacqueline. Mulheres no mercado de trabalho: panorama da década. In: Carreira & Sucesso. *Catho*, 2023. Disponível em: https://www.catho.com.br/carreira-sucesso/mulheres-no-mercado-de-trabalho-panorama-da-decada/. Acesso em: 18 ago. 2023.

DRESSNER, Sarah. Career Ladders: A sample of career ladders I use for my organization, open sourced for anyone. *Career Ladders*, c2023. Disponível em: https://career-ladders.dev/engineering/. Acesso em: 18 ago. 2023.

EFEITO Dunning–Kruger. In: WIKIPÉDIA, a enciclopédia livre. c2022. Disponível em: https://pt.wikipedia.org/wiki/Efeito_Dunning%E2%80%93Kruger. Acesso em:

FERRACCIU, Stefanie. Plano de carreira em Y: o que é, vantagens e mais sobre esse modelo. *Gupy Blog*, 2021. Disponível em: https://www.gupy.io/blog/carreira-em-y. Acesso em: 18 ago. 2023.

FIORANELLI, Jorge. A framework for Engineering Managers. *engineeringladders*, c2022. Disponível em: http://www.engineeringladders.com/. Acesso em: 18 ago. 2023.

GLASSDOOR. *Salários do cargo de Senior Software Engineer – São Paulo*. c2023. Disponível em: https://www.glassdoor.com.br/Sal%C3%A1rios/s%C3%A3o-paulo-senior-software-engineer-sal%C3%A1rio-SRCH_IL.0,9_IM1009_KO10,34.htm. Acesso em: 18 ago. 2023.

HR CONSULTANT UK. *Y career*: understand what it is and its possibilities for professionals. c2021. Disponível em: https://hrconsultantuk.co.uk/y-

career-understand-what-it-is-and-its-possibilities-for-professionals/. Acesso em: 18 ago. 2023.

IBGE. *Pessoas com Deficiência e as Desigualdades Sociais no Brasil.* IBGE: Coordenação de População e Indicadores Sociais. 2022. Disponível em: https://www.ibge.gov.br/estatisticas/sociais/populacao/34889-pessoas-com-deficiencia-e-as-desigualdades-sociais-no-brasil.html?=&t=publicacoes. Acesso em: 18 ago. 2023.

INDEEED. *Tipos de plano de carreira*. 2023. Disponível em: https://br.indeed.com/conselho-de-carreira/desenvolvimento-de-carreira/tipos-plano-carreira. Acesso em: 18 ago. 2023.

LARSON, Will. *Staff Engineer*: Leadership beyond the management track. (s. l.): Will Larson, 2021.

LARSON, Will. Staff archetypes. *StaffEng*, c2023. Disponível em: https://staffeng.com/guides/staff-archetypes/. Acesso em: 18 ago. 2023.

LEVELS.FYI. Disponível em: https://www.levels.fyi/. Acesso em: 18 ago. 2023.

MARTINEZ, Wanise. Por que mulheres se candidatam

a menos vagas de emprego que homens. *TAB Uol*, 2020. Disponível em: https://tab.uol.com.br/noticias/redacao/2020/09/18/por-que-mulheres-se-candidatam-a-menos-vagas-de-emprego-do-que-homens.htm. Acesso em: 18 ago. 2023.

MCDOWELL, GAYLE L. *Cracking the Coding Interview*: 189 Programming Questions and Solutions. Palo Alto/CA: Careercup, 2015.

PACETE, Luiz G. O que é uma startup unicórnio? *Forbes*, 2022. Disponível em: https://forbes.com.br/forbes-money/2022/04/o-que-e-uma-startup-unicornio/. Acesso em: 18 ago. 2023.

PATTON, Bruce; STONE, Douglas; HEEN, Sheila. *Conversas difíceis*: Como discutir o que é mais importante. Rio de Janeiro: Sextante, 2021.

REILLY, Tanya. *The Staff Engineer's Path*: A Guide for Individual Contributors Navigating Growth and Change. Sebastopol: O'Reilly, 2022.

RODRIGUES, Léo. IBGE: renda média de trabalhador branco é 75,7% maior que de pretos. *Agência Brasil*, 2022. Disponível em:

https://agenciabrasil.ebc.com.br/economia/noticia/2022-11/ibge-renda-media-de-trabalhador-branco-e-757-maior-que-de-pretos. Acesso em: 18 ago. 2023.

ROSENBERG, Marshall B. *Comunicação não violenta*: Técnicas para aprimorar relacionamentos pessoais e profissionais. São Paulo: Ágora, 2021.

SÍNDROME do impostor. In: WIKIPÉDIA, a enciclopédia livre. c2023. Disponível em: https://pt.wikipedia.org/wiki/S%C3%ADndrome_do_impostor. Acesso em: 18 ago. 2023.

STONE, Douglas; HEEN, Sheila. *Obrigado pelo feedback*. São Paulo: Portfolio-Penguin, 2016.

T-SHAPED skills. In: WIKIPÉDIA, a enciclopédia livre. c2022. Disponível em: https://en.wikipedia.org/wiki/T-shaped_skills. Acesso em: 18 ago. 2023.

THE GITLAB HANDBOOK. *Distinguished software engineer*. 2023a. Disponível em: https://handbook.gitlab.com/job-families/engineering/development/management/distinguished/. Acesso em: 18 ago. 2023.

THE GITLAB HANDBOOK. *Fellow*. 2023b. Disponível em: https://handbook.gitlab.com/job-families/engineering/development/management/fellow/. Acesso em: 18 ago. 2023.

VALVE. *Handbook For New Employees*. Bellevue: Valve Press, 2012. Disponível em: https://steamcdn-a.akamaihd.net/apps/valve/Valve_Handbook_LowRes.pdf. Acesso em: 18 ago. 2023.

XU, Alex. *System Design Interview*: An insider's guide. Independente: 2020. MENTORSHIP. In: WIKIPÉDIA, a enciclopédia livre. c2023. Disponível em: https://en.wikipedia.org/wiki/Mentorship. Acesso em: 18 ago. 2023.

ZAWADZKI, Patrick. What’s The Difference Between a Junior, Mid-Level, and Senior Developer? *Honeypot*, 2021. Disponível em: https://cult.honeypot.io/reads/difference-between-junior-mid-level-and-senior-developer/. Acesso em: 18 ago. 2023.